Transitions in Dynamic Map Labeling*

Thomas Depian, Guangping Li, Martin Nöllenburg, and Jules Wulms

Algorithms and Complexity Group, TU Wien, Vienna, Austria
thomas.depian@tuwien.ac.at,{guangping, noellenburg, jwulms}@ac.tuwien.ac.at

Abstract

The labeling of point features on a map is a well-studied topic. In a static setting, the goal is to find a non-overlapping label placement for (a subset of) point features. In a dynamic setting, the set of point features and their corresponding labels changes, and the labeling has to adapt to such changes. To aid the user in tracking these changes, we can use morphs, here called *transitions*, to indicate how a labeling changes. Such transitions have not gained much attention yet, and we investigate different types of transitions for labelings of points, most notably *consecutive* transitions and *simultaneous* transitions. We give (tight) bounds on the number of overlaps that can occur during these transitions. When each label has a (non-negative) weight associated to it, and each overlap imposes a penalty proportional to the weight of the overlapping labels, we show that it is NP-complete to decide whether the penalty during a simultaneous transition has weight at most k.

Related Version arXiv:2202.11562

1 Introduction

Maps are ubiquitous in the modern world: from geographic to political maps, and from detailed road networks to schematized metro maps, maps are used on a daily basis. Advances in technology allow us to use digital maps on-the-fly and in a highly interactive fashion, by means of panning, zooming, and searching for map features. Besides changes induced by the user, maps can also change passively, for example automated panning during gps routing, or changing points of interest when visualizing time-varying geospatial (point) data.

Important features on a map are often labeled. Examples of such features are areas (such as countries and mountain ranges), curves (for example roads and rivers), and most importantly points (of interest). The aforementioned interactions force map features and their corresponding labels to change, by appearing, disappearing, or changing position. Instead of swapping between the map before and after such changes, we can use morphs, here called *transitions*, to allow the user to more easily follow changes in map features and labelings. Figure 1 shows why such transitions are important: even for two very similar map labelings, a lot of mental effort can be required to identify the differences.

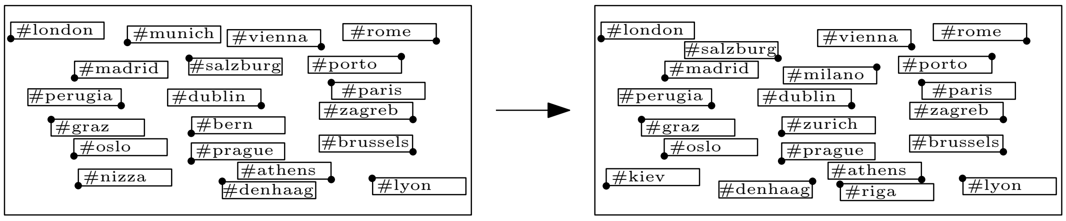


Figure 1 A visual scan of the individual labels is necessary to identify all changes [11].

* GL and JW are (partially) funded by the Austrian Science Fund (FWF) under grant P31119 and JW is partially funded by the Vienna Science and Technology Fund (WWTF) under grant ICT19-035.

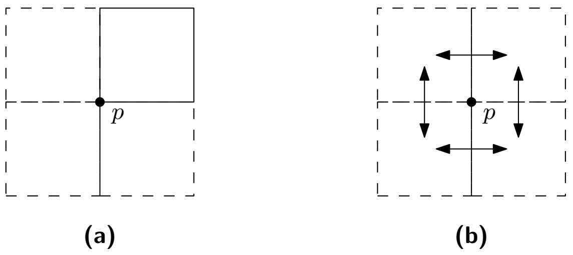


Figure 2 **(a)** The four candidate positions for label l of point p, with l placed in the top-right position. **(b)** Labels continuously move between candidate positions using the sliding-position model.

While previous research focused mainly on (the complexity of) computing labelings, in various static [1, 8, 12], interactive [3, 4, 9, 10], and dynamic [2, 5] settings, in this abstract we study transitions on maps that show point features P and their labels L. Let P be a finite point set in $\mathbb{R}^2$, where each point $p_i \in P$ has a label $l_i \in L$ associated to it. Labels are axis-aligned unit-sized squares in the frequently used four-position model, that is, each point p_i has four possible candidate positions to place label l_i [8] (see Figure 2a). While labels are often modeled as arbitrary (axis-aligned) rectangles, we use squares with side length $\sigma = 1$ for simplicity, and show in [7] how our results extend to arbitrary rectangles. A labeling $\mathcal{L} \subseteq L$ of P consists of a set of pairwise non-overlapping labels, and can be drawn on a map conflict-free, by drawing only the labels that are in $\mathcal{L}$ with their associated points. If the label $l \in L$ for a point $p \in P$ is not contained in $\mathcal{L}$, we do not draw p either.

Furthermore, we work in a dynamic setting, where points appear and disappear at different moments in time, and hence the set P changes through additions and deletions. Every time additions and deletions are made to P, a new overlap-free labeling must be computed, thus resulting in a change from labeling $\mathcal{L}_1$, before the changes, to labeling $\mathcal{L}_2$, afterwards. In this abstract we study different types of transitions from $\mathcal{L}_1$ to $\mathcal{L}_2$. During such a transition, the individual labels are allowed to move in the sliding-position model [12] (see Figure 2b). Our aim is to find transitions that achieve optimization criteria, such as minimizing the number of overlaps during a transition, or minimizing the time required to perform a transition. To our knowledge, this is the first time transitions have been studied in this way.

Problem description. Given two (overlap-free) labelings $\mathcal{L}_1$ and $\mathcal{L}_2$, we denote a transition between them with $\mathcal{L}_1 \rightarrow \mathcal{L}_2$. Such a transition consists of changes of the following types.

Additions If only label l_i of a feature point p_i must be added, we denote this by $\mathcal{L}_1 \xrightarrow{A_i} \mathcal{L}_2$.
Removals If only label l_i of a feature point p_i must be removed, we denote this by $\mathcal{L}_1 \xrightarrow{R_i} \mathcal{L}_2$.
Movements If only label l_i of a feature point p_i must change from its position in $\mathcal{L}_1$ to a new position in $\mathcal{L}_2$, we denote this by $\mathcal{L}_1 \xrightarrow{M_i} \mathcal{L}_2$. Movements are unit speed and axis-aligned, in the sliding-position model. Note that a diagonal movement, as in Figure 3a (left), takes twice as long as a movement to an adjacent position.

A label is *stationary* if it remains unchanged during a transition. Applying multiple transitions consecutively is indicated by chaining the corresponding transition symbols: $\mathcal{L}_1 \xrightarrow{M_i M_j} \mathcal{L}_2$ denotes that label l_i moves before label l_j. Furthermore, $\mathcal{L}_1 \xrightarrow{M} \mathcal{L}_2$ is a shorthand for applying all movement-transitions simultaneously. All these notions extend to additions and removals, using A and R, respectively, instead of M. A transition has no effect if no point must be transformed with the respective transition, e.g., even if there are no additions, the transition $\mathcal{L}_1 \xrightarrow{A} \mathcal{L}_2$ is still applicable; it simply does not modify the labeling.

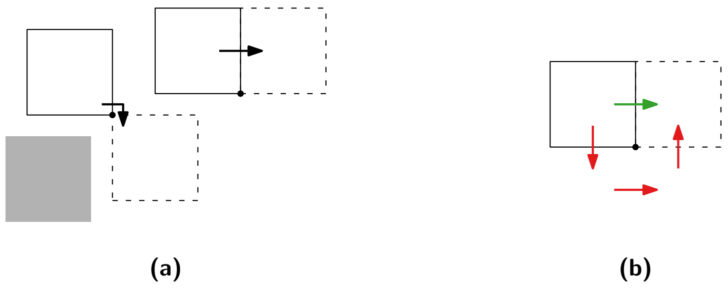


Figure 3 **(a)** Minimizing overlaps by moving around the gray stationary label. **(b)** Minimizing duration by using a single movement along the green arrow, instead of moving along the red arrows.

We aim to identify types of transitions that try to achieve the following goals.

$\mathcal{G}_1$– **Minimize overlaps** While the two labelings are overlap-free, overlaps can occur during the transition from $\mathcal{L}_1$ to $\mathcal{L}_2$. Those overlaps should be avoided as much as possible, by, for instance, adjusting the movement direction of labels, as shown in Figure 3a.

$\mathcal{G}_2$– **Minimize transition duration** The main goal is still to show a map in a (mostly) static state. Hence, we want to perform the transitions as fast as possible. This can be achieved by disallowing detours, as in Figure 3b, or by performing the changes simultaneously.

Optimizing both goals simultaneously is often impossible as there can be a trade-off: performing the transition as fast as possible to achieve $\mathcal{G}_2$ often leads to unnecessary overlaps, while preventing as many overlaps as possible to achieve $\mathcal{G}_1$ may require more time. However, to work towards both $\mathcal{G}_1$ and $\mathcal{G}_2$, we can perform all additions simultaneously, as well as all removals. Furthermore, if we perform removals before movements, and movements before the additions, we create free space for the movements, to reduce the number of overlaps without wasting time. Let X be an arbitrary way of performing all movements required to change from $\mathcal{L}_1$ to $\mathcal{L}_2$ (consecutively and/or simultaneously), then we can observe the following.

▶ **Observation 1.** *A transition of the form* $\mathcal{L}_1 \xrightarrow{RXA} \mathcal{L}_2$ *aids in achieving both* $\mathcal{G}_1$ *and* $\mathcal{G}_2$*.*

In the following sections we introduce and analyze different *transition styles*, each a variant of the style RXA, as prescribed by Observation 1, while filling in X in a unique way.

All omitted proofs and details can be found in the complete version [7].

2 Consecutive Transitions

Naive transitions. Before we can propose more elaborate transition styles, we first evaluate the potential overlaps for a single label performing its movement. Figure 4a shows how only a single stationary square label can interfere with the moving label.

▶ **Lemma 2.1.** *In* $\mathcal{L}_1 \xrightarrow{RM_iA} \mathcal{L}_2$*, where only label* l_i *moves, at most one overlap can occur.*

Next we consider an arbitrary order of all n moving labels in a transition. We define a conflict graph, which has a vertex for each moving label, and an edge between overlapping labels. With a packing argument we locally bound the degree of each of the n moving labels to 14 by considering the start, intermediate, and end position of such a label (these overlaps are achieved in Figure 4b). By the handshaking lemma this results in at most $7n$ overlaps.

▶ **Lemma 2.2.** *In* $\mathcal{L}_1 \xrightarrow{RM_1 \ldots M_nA} \mathcal{L}_2$ *at most* $7n$ *overlaps can occur.*

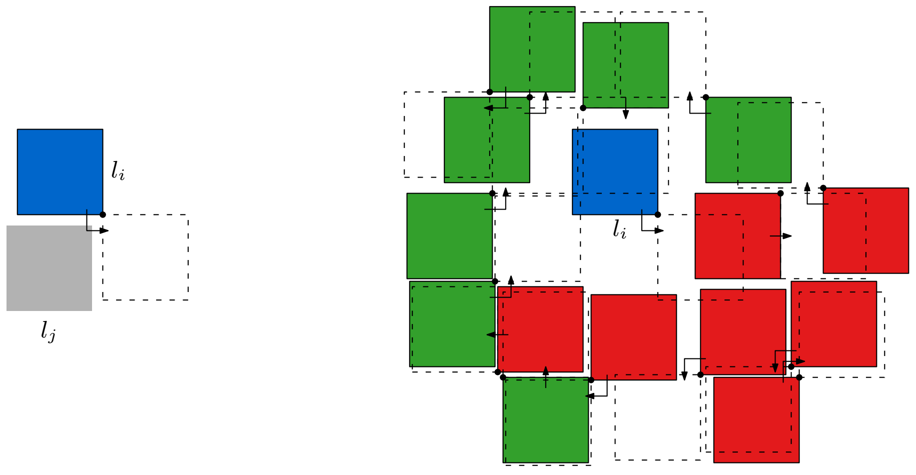


Figure 4 (a) Since all labels are squares with side length σ, the moving blue label l_i can overlap only a single gray stationary label l_j. **(b)** The blue label l_i overlaps 14 other labels during the movement transitions. The green labels move before l_i, red labels move after l_i.

DAG-based transitions. To refine the naive approach, we model dependencies between movements in a *movement graph*, and use it to order movements and avoid certain overlaps.

▶ **Definition 2.3** (Movement graph). Let $\mathcal{M} = \{M_1, \ldots, M_n\}$ be a set of movements. Create for each movement $M_i \in \mathcal{M}$ a vertex v_i, and create a directed edge from v_i to v_j, $v_i \to v_j$, if some intermediate or end position of M_j overlaps with the start position of M_i, or the end position of M_j overlaps with some intermediate position of M_i. If intermediate positions of M_i and M_j overlap, create the edge $v_i \to v_j$, $i < j$. This results in the movement graph $G_{\mathcal{M}}$.

An example for a movement graph is shown in Figure 5.

▶ **Theorem 2.4.** *Movements in $\mathcal{L}_1 \xrightarrow{RM_1 \ldots M_n A} \mathcal{L}_2$ can be rearranged such that at most $n+m$ overlaps occur, if removing m edges transforms $G_{\mathcal{M}}$, with $\mathcal{M} = \{M_1, \ldots, M_n\}$, into a DAG.*

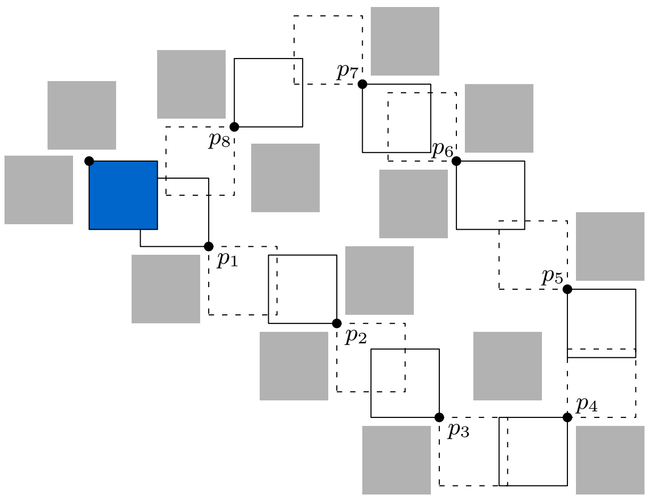
Proof. By Lemma 2.1, we know that at most one overlap occurs when moving a single label to a free end position. This leads to at most n overlaps for n consecutively moving labels, if no label moves to (or through) a position occupied by a label, which starts moving later.

Let $G_{\mathcal{M}}$ be a movement graph with $\mathcal{M} = \{M_1, \ldots, M_n\}$. There are two cases:

Case (1) If $G_{\mathcal{M}}$ is acyclic, then handling all movements according to any topological ordering of the vertices of $G_{\mathcal{M}}$ produces no additional overlaps.

Case (2) If $G_{\mathcal{M}}$ contains cycles, then overlaps may be inevitable because each label in such a cycle wants to move to or through a position that is occupied by another moving label. Moreover, as the movements happen sequentially, one label in this cycle must move first and therefore may cause an overlap. Let m be the smallest number of edges that must be removed to break each cycle in $G_{\mathcal{M}}$, i.e., the size of a minimum feedback arc set S. As $G_{\mathcal{M}}$ is cycle-free after removing S, case (1) applies and m additional overlaps suffice. ◀

We can see in Figure 5 that this bound is tight. Furthermore, it is not always necessary to perform all movements consecutively. We can observe that movements which are unrelated in $G_{\mathcal{M}}$ can be performed simultaneously: when no overlap is possible, there is no edge in $G_{\mathcal{M}}$.



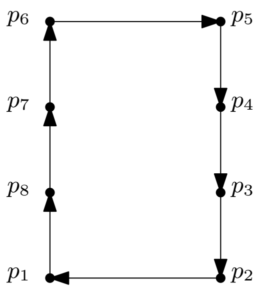


Figure 5 (a) The blue label is added in this transition and forces $n+m$ inevitable overlaps during movement ($n = 8$ and $m = 1$). Gray labels are stationary. **(b)** The corresponding movement graph.

3 Simultaneous Transitions

Figure 6 shows three timelines of different transition styles, (1) a naive consecutive transition, (2) a DAG-based transition, and (3) simultaneous movement. While (1) produces four overlaps and takes four units of time, (2) and (3) produce no overlaps, and (3) only takes a single unit of time. This shows that it is sometimes unnecessary to perform the movements consecutively to minimize overlaps. In this section, we investigate how simultaneous movements influence the number of overlaps, and the complexity of minimizing overlaps.

▶ **Theorem 3.1.** *In $\mathcal{L}_1 \xrightarrow{RMA} \mathcal{L}_2$ at most $6n$ overlaps can occur, where n is the number of labels that must be moved, and all movements are performed at unit speed.*

Proof sketch. We again use a conflict graph, as for Lemma 2.2, with a more intricate packing argument than before (see Figure 7). We consider a σ-wide area around the movement of each label l, and argue where the start positions of labels overlapping l can be located inside this area. We then bound the degree of each of the n moving labels to 12 (and this degree is achieved in Figure 7d), which by the handshaking lemma results in at most $6n$ overlaps. ◀

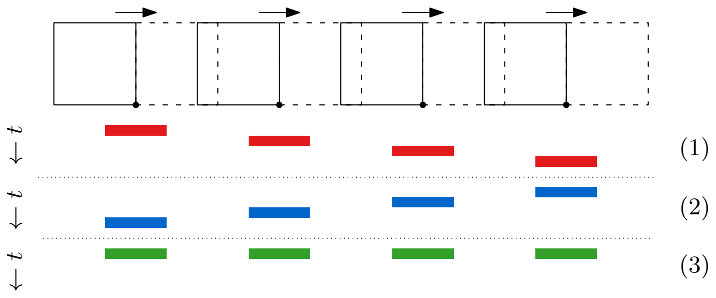


Figure 6 Comparison of possible movement orderings with respect to $\mathcal{G}_1$ and $\mathcal{G}_2$.

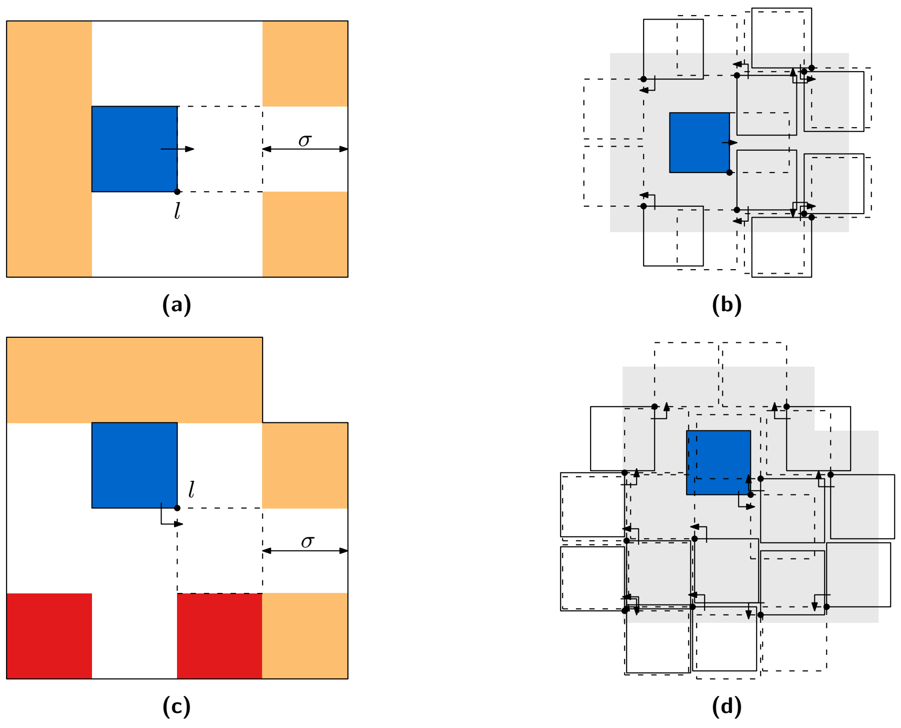


Figure 7 Overlapping regions for **(a)** non-diagonal and **(c)** diagonal movement of the blue label l. Label l has at most **(b)** eight overlaps, **(d)** twelve overlaps with moving (white) labels. Labels starting in orange/red areas cannot overlap l, as l moves away, or they overlap the end position of l.

3.1 Complexity of Computing Simultaneous Transitions

In this section, we show that it is NP-complete to minimize the number of overlaps in a *weighted* $\mathcal{L}_1 \xrightarrow{RMA} \mathcal{L}_2$-transition by choosing the direction of diagonal movements.

▶ **Definition 3.2** (Weighted Transition). Let $\mathcal{L}_1 \xrightarrow{\Sigma} \mathcal{L}_2$ be a transition, where Σ denotes an arbitrary transition style of additions, movements, and removals, and let w be a weight function that assigns to each label $l \in L$ a non-negative weight $w(l) \in \mathbb{R}_0^+$. A weighted transition $\mathcal{L}_1 \xrightarrow[w]{\Sigma} \mathcal{L}_2$ performs $\mathcal{L}_1 \xrightarrow{\Sigma} \mathcal{L}_2$, but when two labels l_i and l_j overlap, a penalty of weight $w(l_i) \cdot w(l_j)$ is introduced. The *total penalty* W is equal to the sum of penalty weights.

▶ Problem 1. Given a weighted transition $\mathcal{L}_1 \xrightarrow[w]{RMA} \mathcal{L}_2$ and $k \in \mathbb{R}_0^+$, can we assign a movement direction to each diagonal movement such that the total penalty W is at most k?

▶ **Theorem 3.3.** *It is NP-complete to decide whether W is at most k for $\mathcal{L}_1 \xrightarrow[w]{RMA} \mathcal{L}_2$.*

Proof sketch. Given a movement direction for each label, it is easy to check whether W is at most k by considering each pair of labels and checking for overlaps. Hence Problem 1 is contained in NP. For NP-hardness, we reduce from an instance F of Planar Monotone Max 2-Sat [6]. Figure 8 gives an overview of the required gadgets. Clause and variable gadgets consist of two opposing labels at their core, corresponding, respectively, to the assignments of the two literals in a clause, or the binary choice for a variable. For an unsatisfied clause, an overlap occurs inside the clause gadget, whenever both labels move towards each other (inwards). The corresponding labels have weight one, and hence such

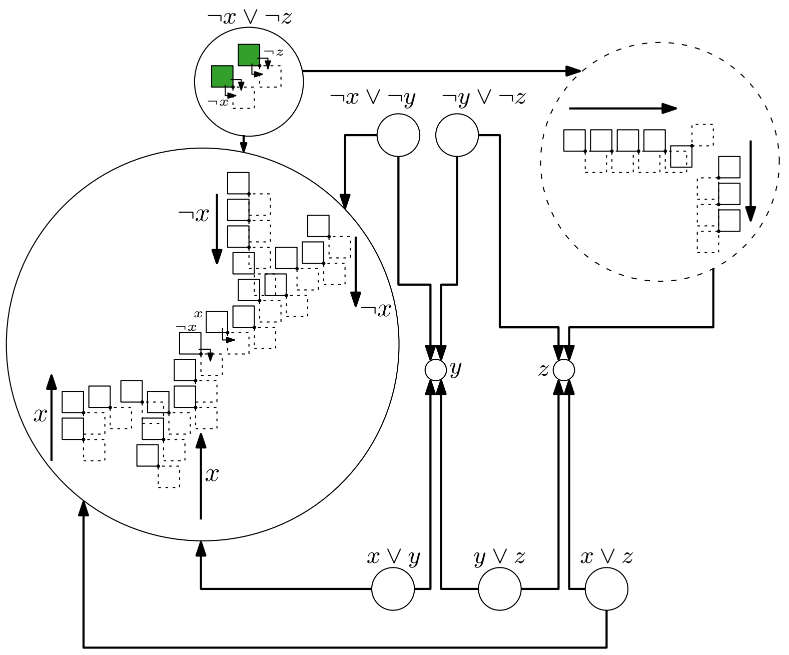


Figure 8 Reduced instance for the formula $F = (\neg x \vee \neg z) \wedge (\neg x \vee \neg y) \wedge (\neg y \vee \neg z) \wedge (x \vee y) \wedge (y \vee z) \wedge (x \vee z)$. The weight of white and green labels is $n + 1$ and 1, respectively.

an overlap would incur a penalty of weight one. A variable gadget has two opposing labels for setting the variable to *true* or *false*. Choosing a movement direction outward from the variable gadget, for example on the "true"-side, will cause a domino effect, propagating towards the gadgets of clauses with negative occurrences of this variable. There it results in inward movement, and hence this corresponds to setting the variable to not be false (and thus be true). Choosing the outward movement for both variable states is never beneficial: that variable is neither true nor false. The movement directions chosen in the variable gadgets are propagated to the appropriate clauses using the (planar) embedding of the incidence graph of F. All labels outside of clause gadgets have weight $n + 1$ and hence producing an overlap outside of a clause gadget will result in a large penalty of weight greater than n. As such, we either have movement directions that produce a total penalty of at most k for some positive $k < n$, and overlaps correspond to unsatisfied clauses, or we have a total penalty of at least n, and no clauses can be satisfied (or the variable assignment is inconsistent). Thus, $n - k$ clauses are satisfiable in F, if and only if we have k overlaps in our reduced instance. ◀

4 Conclusion

In this abstract we performed a first investigation into the number of overlaps produced by transitions on labelings of points, and started by proving tight upper bounds for various transition styles. Finally, we showed that it is NP-complete to decide whether a weighted simultaneous transition has a penalty of at most k. We see this abstract as a first step towards understanding such transitions in map labeling. Therefore we have many open questions for future work, such as:

- Do transitions work well in practice? Can we verify our results with a prototype?

- Should we develop new transition styles or improve the existing ones? Can we utilize more structured movement, like performing all movements in the same direction simultaneously?
- Is choosing the direction of labels in simultaneous transitions still NP-hard in the unit weight case?
- Can we analyze transitions from the point of view of (algorithmic) stability?

References

1 Pankaj K. Agarwal, Marc J. van Kreveld, and Subhash Suri. Label placement by maximum independent set in rectangles. *Computational Geometry*, 11(3-4):209–218, 1998. doi:10.1016/S0925-7721(98)00028-5.

2 Lukas Barth, Benjamin Niedermann, Martin Nöllenburg, and Darren Strash. Temporal map labeling: a new unified framework with experiments. In *Proc. 24th ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, pages 1–10, 2016. doi:10.1145/2996913.2996957.

3 Ken Been, Eli Daiches, and Chee-Keng Yap. Dynamic map labeling. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):773–780, 2006. doi:10.1109/TVCG.2006.136.

4 Ken Been, Martin Nöllenburg, Sheung-Hung Poon, and Alexander Wolff. Optimizing active ranges for consistent dynamic map labeling. *Computational Geometry*, 43(3):312–328, 2010. doi:10.1016/j.comgeo.2009.03.006.

5 Sujoy Bhore, Guangping Li, and Martin Nöllenburg. An Algorithmic Study of Fully Dynamic Independent Sets for Map Labeling. In *Proc. 28th European Symposium on Algorithms (ESA)*, volume 173 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 19:1–19:24, 2020. doi:10.4230/LIPIcs.ESA.2020.19.

6 Kevin Buchin, Valentin Polishchuk, Leonid Sedov, and Roman Voronov. Geometric Secluded Paths and Planar Satisfiability. In *Proc. 36th International Symposium on Computational Geometry (SoCG)*, volume 164 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:15, 2020.

7 Thomas Depian, Guangping Li, Martin Nöllenburg, and Jules Wulms. Transitions in Dynamic Map Labeling, 2022. doi:10.48550/arXiv.2202.11562.

8 Michael Formann and Frank Wagner. A packing problem with applications to lettering of maps. In *Proc. 7th International Symposium on Computational Geometry (SoCG)*, pages 281–288, 1991.

9 Andreas Gemsa, Martin Nöllenburg, and Ignaz Rutter. Consistent Labeling of Rotating Maps. *Journal of Computational Geometry*, 7(1):308–331, 2016. doi:10.20382/jocg.v7i1a15.

10 Chung-Shou Liao, Chih-Wei Liang, and Sheung H. Poon. Approximation algorithms on consistent dynamic map labeling. *Theoretical Computer Science*, 640:84–93, 2016. doi:10.1016/j.tcs.2016.06.006.

11 Ronald A. Rensink, John K. O'Regan, and James J. Clark. To See or not to See: The Need for Attention to Perceive Changes in Scenes. *Psychological Science*, 8(5):368–373, 1997. doi:10.1111/j.1467-9280.1997.tb00427.x.

12 Marc J. van Kreveld, Tycho Strijk, and Alexander Wolff. Point labeling with sliding labels. *Computational Geometry*, 13(1):21–47, 1999. doi:10.1016/S0925-7721(99)00005-X.