Optimizing Active Ranges for Consistent Dynamic Map Labeling

Ken Been[a], Martin Nöllenburg[b,1], Sheung-Hung Poon[c,2], Alexander Wolff[d]

[a] *Cyrus Innovation, LLC, New York, NY, U.S.A.*
[b] *Faculty of Informatics, Universität Karlsruhe and Karlsruhe Institute of Technology (KIT), Germany*
[c] *Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, R.O.C.*
[d] *Faculteit Wiskunde en Informatica, TU Eindhoven, The Netherlands*

Abstract

Map labeling encounters unique issues in the context of dynamic maps with continuous zooming and panning—an application with increasing practical importance. In *consistent* dynamic map labeling, distracting behavior such as popping and jumping is avoided. We use a model for consistent dynamic labeling in which a label is represented by a 3d-solid, with scale as the third dimension. Each solid can be truncated to a single scale interval, called its *active range*, corresponding to the scales at which the label will be selected. The *active range optimization (ARO)* problem is to select active ranges so that no two truncated solids intersect and the sum of the heights of the active ranges is maximized. *Simple* ARO is a variant in which the active ranges are restricted so that a label is never deselected when zooming in. We investigate both the general and simple variants, for 1d- as well as 2d-maps.

Different label shapes define different ARO variants. We show that 2d-ARO and general 1d-ARO are NP-complete, even for quite simple shapes. We solve simple 1d-ARO optimally with dynamic programming, and present a toolbox of algorithms that yield constant-factor approximations for a number of 1d- and 2d-variants.

Key words: consistent dynamic map labeling, active range optimization, NP-hardness, approximation algorithms

1. Introduction

Recent years have seen tremendous improvements in Internet-based, geographic visualization systems that provide continuous zooming and panning, but relatively little attention has been paid to special issues faced by map labeling in such contexts. In addition to the need for interactive speed, several desiderata for a *consistent dynamic labeling* were identified by Been et al. [2]: labels should not pop in and out or jump (suddenly change position or size) during panning and zooming, and the labeling should not depend on the user's navigation history. Currently available systems (for example, Google Earth, NASA World Wind, Microsoft Virtual Earth, and KDE Marble) do not satisfy these desiderata and their labeling algorithms may produce rather unattractive dynamic labelings—at least during user interaction.

In *static labeling* we want to select a (maximum) subset of labels that can be placed without intersection, given certain constraints on the size, location and orientation of each label. A natural extension of this to dynamic labeling is to select *at each scale* a maximum subset of labels that can be placed without intersection, subject to similar constraints on the label placements. We take the aforementioned consistency desiderata as additional constraints. Clearly the desiderata mandate that the labeling at scale s must take into account labelings at scales in the neighborhood of s. In fact we can go further: by taking scale as our "vertical" dimension, the desiderata mandate that the label placement must be continuous with scale—that is, the

Email addresses: kbeen@cyrusinnovation.com (Ken Been), spoon@cs.nthu.edu.tw (Sheung-Hung Poon)
URL: i11www.iti.uka.de/group/noelle (Martin Nöllenburg), www.win.tue.nl/~awolff (Alexander Wolff)
[1] Supported by grant WO 758/4-3 of the German Research Foundation (DFG).
[2] Supported by grant 97-2218-E-007-006 of the National Science Council (NSC), Taiwan, R.O.C.

(a) Input: a set of extrusions.

(b) Output: a set of disjoint truncated extrusions.

Figure 1: The active range optimization problem.

dynamic placement of a two-dimensional label becomes a three-dimensional solid formed by extruding the label shape through the scale dimension, possibly with a continuous morph and a continuous change in position. Figure 1a shows a simple example with three one-dimensional labels and scale as the vertical dimension.

Dynamic label "solids", as in Figure 1a, define the *potential* placement of each label—in other words, its location, shape, and size—at any scale at which the label is selected. The next step in labeling, and the subject of this paper, is to determine at which scales to select each label. Once again, the consistency desiderata constrain and simplify our problem: since there can be no popping during zooming, each label must be selected (if at all) on a single range of scales, which we call its *active range*. Figure 1b shows a selection of an active range for each label, so that no two selected labels intersect at any scale. The *active range optimization* (ARO) problem is to select a set of active ranges that maximizes the sum of the heights of these ranges. Clearly this sum of heights is equivalent to the integral, over all scales, of the number of labels selected at each scale, and so this optimization is equivalent to our original goal of maximizing the number of labels selected at each scale.

Model. A highly detailed model for dynamic labeling is given by Been et al. [2]. The model we present here is similar in spirit but significantly reformulated, with many of the details abstracted out.

In *static labeling* the key operations are selection and placement—select a subset of the labels that can be placed without intersection. Let each label L be defined in its own label coordinates. A *static placement* of L is its image $\hat{L}$ in world coordinates under a transformation composed of translation, rotation, and scaling (see Figure 2). To clearly distinguish from "map scale", we refer to the scaling operation as *dilation*. A

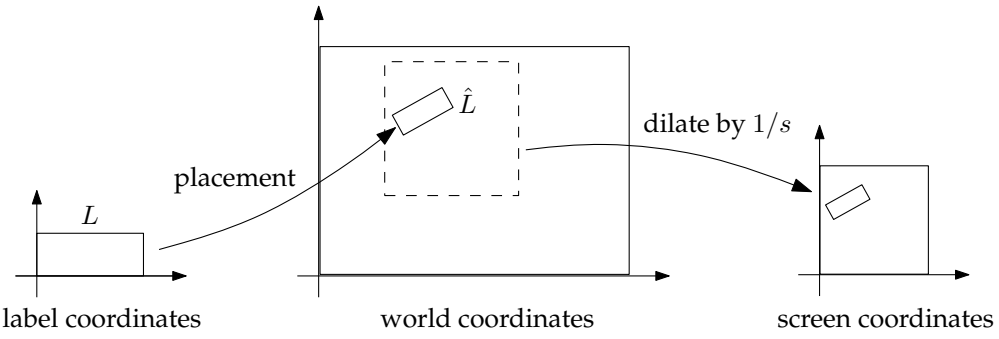


Figure 2: Label, world, and screen coordinates.

further transformation takes a portion of world coordinates to the screen, dilating by factor $1/s$; we define s to be the *map scale*, or just *scale*. Note that s is the inverse of cartographic scale, and that if $s < 1$ world coordinates are magnified on screen; for maps in cartography we generally have $s \gg 1$.

We define *extended world coordinates* by adding a scale dimension to world coordinates. We define a *dynamic placement* of a label L to be a solid $E = E(L)$ in extended world coordinates formed by sweeping a rotated and dilated image of L along a continuous curve segment that is monotonic in scale, with the

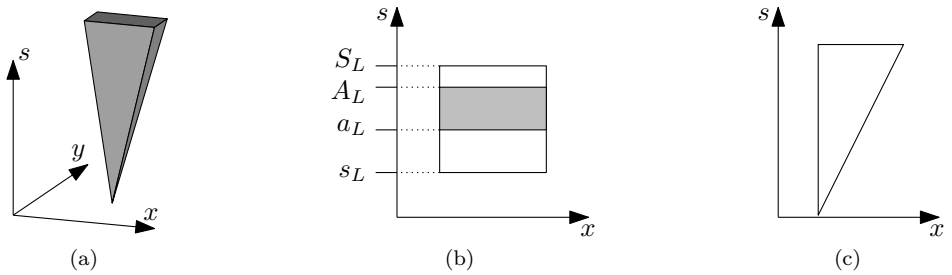


Figure 3: (a) A dynamic placement of a 2d-label is a solid in extended world coordinates. Here: with proportional dilation and invariant point placement with the center as the invariant point. (b) A 1d-label with constant dilation, selectable range (s_L, S_L), and active range (a_L, A_L). (c) A 1d-label with proportional dilation and the left endpoint of the label as the invariant point.

rotation and dilation factors given by continuous functions of scale. We call E the *extrusion* of L. An example is given in Figure 3a. Let the lower and upper endpoints of the sweep curve segment have scale s_L and S_L, respectively. Let the *trace* $\mathrm{tr}_{s^\star}(E)$ of E at scale $s^\star$ be the intersection of E with the horizontal line or plane $s = s^\star$ for 1d and 2d labels, respectively. Then $\mathrm{tr}_s(E)$ is a static placement of L for each $s \in (s_L, S_L)$.

We define the *selectable range* of L to be the open interval (s_L, S_L). The selectable range defines the scales at which L is available for selection. For example, street labels are selectable at smaller scales and country labels at larger scales. Let $S_{\max}$ be a universal maximum scale for all labels. Then we require that $(s_L, S_L) \subseteq (0, S_{\max})$.

We define *dynamic selection* to be a Boolean function of scale. We require that each label L be selected precisely on a single interval of scales, $(a_L, A_L) \subseteq (s_L, S_L)$, which is called the *active range*[3] of L. It defines the scales at which L is actually selected or *active*. We define the *truncated extrusion* of L to be the restriction of E to the active range (a_L, A_L). If its active range is empty then L is never selected.

This model is quite general. The extrusion shapes are determined by the label shape and the translation, rotation and dilation functions that compose the dynamic placement, the only restriction being that these functions are continuous. For the specific ARO variants that we consider in this paper, however, we now restrict our attention to certain classes of extrusions that are simple and yet arise naturally in applications.

Our 2d-labels are open rectangles (for example, bounding boxes of textual labels); we also consider 1d-labels, which are open intervals on the x-axis—see Figure 3b. For rotation and translation, we consider only *axis-aligned invariant point placements*: the rotation component is constant and maps L to an axis-aligned rectangle at each scale, and the translation component is constant and maps a particular reference point of the label always to the same location in world coordinates. In other words, our labels never slide or rotate. For the dilation component we consider only linear functions of the form $D^L(s) = bs + c$, and only three classes of these:

- If $b = 0$ and $c > 0$, that is, $D^L(s) = c$, then the label size is fixed in world coordinates and inversely proportional to scale on screen. Thus labels shrink at the same rate as the geographic features when zooming out and grow when zooming in. The solid is then a "straight" extrusion, as in Figure 3b.
- If $b > 0$ and $c = 0$, that is, $D^L(s) = bs$, then L has constant size on screen and size proportional to scale in world coordinates. The solid is then a label-shaped cone with apex at $s = 0$ as in Figure 3a. With invariant point placements, the cone contains the vertical line through its apex. The cone might

[3]For visualization we should consider the active range of L a half-open interval $(a_L, A_L]$ since this gives a seamless transition from one label to the next during zooming. For notational convenience, simplicity, and symmetry, however, we use open intervals. This does not change our results.

be symmetric to that line (for example, for labeling a region) as in Figure 3a, or it might be slanted and have this line as a vertical corner edge (for example, for labeling a point) as in Figure 3c.

- If $D^L(s) = bs + c$ for constants $b > 0$ and $c \neq 0$, then label size is disproportionate to scale in world coordinates and on screen. The solid in this case is a portion of a label-shaped cone with apex at $-c/b$.

Objective. In principle, our input is a set of labels, each with a selectable range, for which we want to compute suitable active ranges. Since our results in this paper, however, are based on the shape of the corresponding extrusions and their intersection behavior, whereas the underlying labels are used only to define the extrusions, it simplifies our notation to consider from now on the set $\mathcal{E}$ of induced extrusions as the input. In this regard, we refer to the selectable and the active range of the extrusion $E = E(L)$ of a label L as the intervals $(s_E, S_E) = (s_L, S_L)$ and $(a_E, A_E) = (a_L, A_L)$, respectively.

So assume we are given a set $\mathcal{E}$ of extrusions, each with a selectable range. We will compute an assignment $\mathcal{A}$ of active ranges to $\mathcal{E}$, that is, for each extrusion $E \in \mathcal{E}$ an interval $(a_E, A_E) \subseteq (s_E, S_E)$. We say that $\mathcal{A}$ is *valid* (or a *solution*) if the resulting truncated extrusions are pairwise disjoint. For a solution $\mathcal{A}$ we say that an extrusion E is *active in $\mathcal{A}$ at scale s* if $s \in (a_E, A_E)$; otherwise E is *inactive in $\mathcal{A}$ at scale s*. If the active range of E in $\mathcal{A}$ is empty, we stipulate $a_E = A_E$ and say that E is *inactive in $\mathcal{A}$*; otherwise E is *active in $\mathcal{A}$*. Define the *total active range height* of $\mathcal{A}$ to be $H(\mathcal{A}) = \sum_{E \in \mathcal{E}} (A_E - a_E)$. This is the same as integrating over all scales a function that counts the number of labels selected at scale s. The *(general) active range optimization (ARO)* problem is to choose the active ranges so as to maximize H, subject to the constraint that $\mathcal{A}$ is valid. It is of theoretical and practical interest, for example in technical maps showing measurements, to also consider a version of the problem in which all labels are selectable at all scales and a label is never deselected when zooming in—that is, $(s_E, S_E) = (0, S_{\max})$ and $a_E = 0$ for all $E \in \mathcal{E}$. We call this variant of ARO *simple*.

Although we concentrate solely on maximizing H, other objectives are possible. One may want to maximize the average percentage that each label uses of its selectable range. This objective, however, has the disadvantage that every label must be placed at some scale range. Maximizing the minimum active range height does not seem useful either since a few labels could have a very small selectable range.

Related problems. Consider the special case of 1d-ARO where all extrusions are axis-aligned unit squares. We do not know the complexity of this packing problem, but we give an efficient (2/3)-approximation algorithm (Theorem 7) and we do show that the problem is hard if the squares can choose between two sizes (Theorem 1). Clearly, the problem is closely related to geometric maximum independent set problems, that is, maximum independent set problems in intersection graphs of geometric objects. Such problems are usually NP-hard and admit—if the geometric representation is given—polynomial-time approximation schemes, for example in the case of axis-aligned unit squares [9], even if the size restriction is dropped [7]. In those problems, however, one can choose only to put or not put a complete object into the solution, whereas in ARO one can put an arbitrarily small fraction of an object into the solution. We insist only that each object contributes at most one connected component to the solution and that such a component must fill the whole width of the object.

The above-mentioned special case of 1d-ARO can also be viewed as a scheduling problem with geometric constraints. To see this, use *line stabbing* [1]: stab the unit squares with vertical lines of distance greater than 1 such that each square is stabbed and each line stabs a square. Now each stabbing line corresponds to a machine, each square corresponds to a job that is run on the corresponding machine, and the y-axis is the time axis. Each machine executes at most one job at a time, and a job can be started at most once. While a job is being processed, it blocks other jobs if the corresponding squares intersect.

Finally, there is some similarity to dynamic storage allocation, where one gets requests for blocks of contiguous bytes of memory. Each request has a start and an end time. In this problem, the x-axis is the time axis and the positive integers on the y-axis correspond to memory cells. A request corresponds to a rectangle of fixed size that can slide vertically. Each request must be placed, and the aim is to minimize the amount of memory that has to be allocated, that is, the smallest y-coordinate occupied by a

d	ARO	extrusion shape	dilation	NPC	approx.	running time $O(\cdot)$	see
1	simple	triangles	bs	no	optimal	n^3	Thm. 4
	general	unit squares	c	?	2/3	$n \log n$	Thm. 7
		unit-height rectangles	c	?	1/3	$n \log n$	Thm. 5
		unit-width rectangles	c	?	1/2	$n \log n$	Thm. 6
		rectangles	c	yes	$1/\log n$	$n \log n$	Thm. 8
		segments of congruent triangles	bs	?	1/2	$(k+n) \log n$	Thm. 10
		congruent trapezoids	$bs + c$	?	1/2	$(k+n) \log n$	Thm. 9
2	simple	congruent square cones	bs	yes	1/4	$(k+n) \log^2 n$	Cor. 1
		congruent square cones	bs	yes	$1/4 - \varepsilon$	$n \log n \cdot \log(n/\varepsilon)/\varepsilon$	Thm. 14
		arbitrary square cones	bs	yes	1/24	$n \log^3 n$	Thm. 13
	general	segments of congruent square cones	bs	yes	1/4	$(k+n) \log^2 n$	Thm. 12
		congruent square frusta	$bs + c$	yes	$1/(4W)$	n^2	Thm. 11

Table 1: Results attained in this paper, where d is the dimension of the ARO problems, NPC means NP-complete, k is the number of pairwise intersections between (side edges/faces of) extrusions, $\varepsilon > 0$, and W is the width ratio of top over bottom side.

rectangle. Buchsbaum et al. [3] give a $(2+\varepsilon)$-approximation algorithm for this problem and polynomial-time approximation schemes for a number of special cases.

Previous work. Map labeling has been identified as an important application area by the Computational Geometry Impact Task Force [4], and has been the focus of extensive algorithmic investigation [17]. The vast majority of research on this topic covers *static* labeling. A typical goal is to select and place labels without intersection while optimizing an objective function. The objective function might be simply the number of labels [1, 16], or it might incorporate multiple cartographic criteria [5]. There are many variations possible, and most have been shown to be NP-hard [8, 10, 16].

For *dynamic* labeling, Petzold et al. [13, 14] use a preprocessing phase to generate a data structure that is searched during interaction to produce a labeling for the current scale and view area. Popping can occur because a small change in scale or location leads to a recomputation of the labeling on a different set of labels. Poon and Shin [15] precompute solutions for a number of scales; interpolating between these yields solutions for any scale. Popping and jumping effects can occur because during zooming the labeling solution for the current scale is computed regardless of the solution for the previous scale. In addition to introducing consistency for dynamic map labeling, Been et al. [2] show that simple 2d-ARO is NP-complete for arbitrary star-shaped labels, and implement a simple heuristic solution.

Outline. We investigate the complexity of ARO in Section 2. We prove that general 1d-ARO with constant dilation is NP-complete, even if all extrusions are squares, and that simple 2d-ARO with proportional dilation is NP-complete, even if all extrusions are congruent square cones. Both proofs are by reduction from Planar 3SAT, the latter using 3d gadgets. We present an algorithmic study of ARO in Section 3, developing a toolbox of techniques to solve several variants of ARO problems. One variant is solved exactly, the others with approximations. Table 1 summarizes our results. Note that all our results for 2d-ARO with congruent square cones can be generalized to congruent and non-rotated rectangular cones by dilating the input space in x- or y-direction. Similarly, the result for arbitrary square cones can be generalized to similar, non-rotated rectangular cones.

Dynamic labeling is a new sub-discipline of map labeling and our work presents a first extensive collection of complexity and algorithmic results. Many questions, however, remain unsolved—most notably: does any of our problems have a polynomial-time approximation scheme?—and require future effort. We conclude the paper with a list of open problems in Section 4.

2. Complexity

In this section, we prove that two variants of ARO are NP-complete, namely general 1d-ARO with constant dilation (see Section 2.1) and simple 2d-ARO with proportional dilation (see Section 2.3). We also show that we can reduce a variant of 1d-ARO with constant dilation to 1d-ARO with proportional dilation, see Section 2.2.

Both hardness proofs use a reduction from the NP-hard problem Planar 3SAT [11]. An instance of Planar 3SAT is a Boolean 3SAT formula φ whose variable-clause graph G_φ is planar. Note that G_φ can be laid out (in polynomial time) such that variables correspond to rectangles centered on the x-axis and clauses correspond to non-crossing three-legged "combs" completely above or completely below the x-axis [10], see Figure 4.

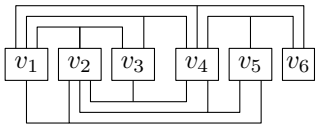


Figure 4: Layout of a planar 3SAT formula.

2.1. General 1d-ARO with constant dilation for rectangles

Theorem 1. *General 1d-ARO with constant dilation is NP-complete; that is, given a set $\mathcal{E}$ of n axis-aligned rectangular extrusions in the plane and a real number $K > 0$, it is NP-complete to decide whether there is a valid assignment $\mathcal{S}$ of active ranges to $\mathcal{E}$ with $H(\mathcal{S}) \geq K$. The problem remains NP-complete when restricted to instances where all extrusions are squares and each has one of two sizes.*

Proof. For membership in $\mathcal{NP}$, decompose each $E \in \mathcal{E}$ into $O(n)$ horizontal strips determined by the lines $\{s = s_E, s = S_E \mid E \in \mathcal{E}\}$. It is not hard to see that there is an optimal solution that corresponds to a union of such strips. So we can guess a subset of the strips and then check in polynomial time whether (a) all strips from the same square are consecutive, (b) the strips are pairwise disjoint, and (c) their total height is at least K.

To show hardness, let φ be an instance of Planar 3SAT. We construct a set $\mathcal{E}_\varphi$ of squares as illustrated in Figure 5 and fix a threshold $K > 0$ such that $H(\mathcal{S}) \geq K$ for an optimal solution $\mathcal{S}$ if and only if φ is satisfiable.

The squares in $\mathcal{E}_\varphi$ have side length 1 or 5. We refer to a square of side length j as a j-square. A *regular* 5-square is a 5-square that contains a vertically and horizontally centered chain of four disjoint 1-squares. Thus the 5-square can contribute at most two units to H if the 1-squares contribute one unit each. This is a total of six units for the group of five squares, which is more than if the 5-square contributed five units and the 1-squares zero. Geometrically, the contribution of a 5-square is a (5×2)-rectangle that can either appear above or below the chain of 1-squares. We say that the 5-square is in upper or lower *state*, which gives us a means to encode Boolean values. The contributing part of each square is shaded in Figure 5.

Each square in $\mathcal{E}_\varphi$ belongs to a variable gadget, a literal gadget, or a clause gadget. These gadgets correspond one-to-one to the n variables, $3m$ literals, and m clauses of φ, respectively.

Variable gadgets. The gadget of a variable x consists of a horizontal chain of n_x regular 5-squares, where n_x is proportional to the number of times that x appears in φ. Two adjacent 5-squares are joined by two *connectors*, that is, pairs of 1-squares as depicted in Figure 5. Each 1-square of a connector intersects both the adjacent 5-square and the other 1-square in the connector. Since the two 1-squares of a connector intersect, a connector can contribute at most one unit in total. Moreover, if two adjacent 5-squares were in the same state, one of the connectors would contribute zero. Therefore, in any optimal solution for a variable gadget the states of adjacent 5-squares alternate. We let x being *true* correspond to the leftmost 5-square of the gadget being in upper state. Summing up yields that the total contribution of the gadget of x to H is $6n_x + 2(n_x - 1) = 8n_x - 2$ units, irrespective of its truth state.

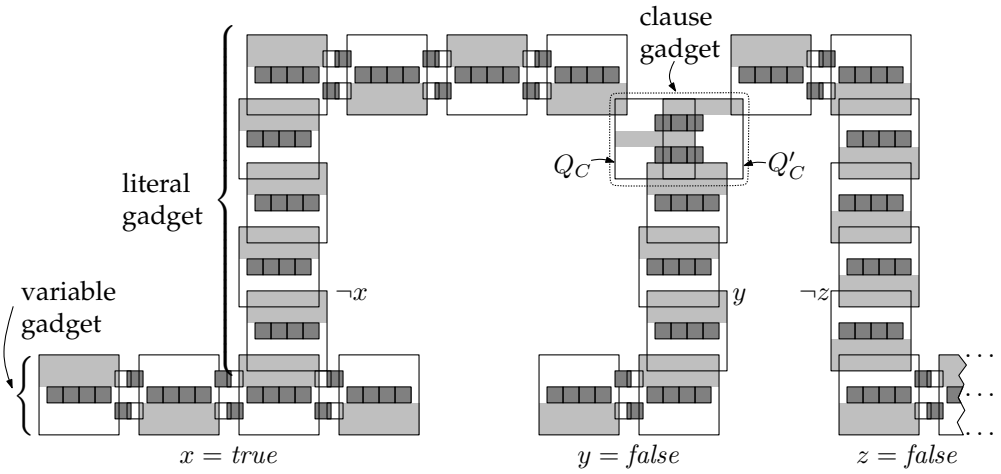


Figure 5: The gadgets of our reduction for the clause $C = (\neg x \vee y \vee \neg z)$.

Literal gadgets. A clause of φ consists of three literals. A literal gadget connects a variable gadget to a clause gadget, implementing one of the three legs of the aforementioned comb. The gadget of a literal λ consists of a vertical part and, if λ corresponds to the left or right leg of a comb, a horizontal part. A vertical part consists of a chain of regular 5-squares where consecutive squares overlap by one unit. A horizontal part is identical to a variable gadget, see Figure 5. The last square of the vertical part is the first square of the horizontal part.

Number the 5-squares in each variable gadget from left to right. If λ is negated, the first 5-square of the gadget of λ overlaps the top of an odd-numbered (or the bottom of an even-numbered) 5-square of the corresponding variable gadget by one unit. Otherwise parity flips; see the positions of the gadgets of literals $\neg x$ and y in Figure 5.

Note that the vertical part of a literal gadget contributes maximally to H (that is, with six units per regular 5-square) if all 5-squares are in the same state as the intersected 5-square of the variable gadget.

If the gadget of a literal λ has a horizontal part, then the states of the 5-squares in that horizontal part alternate as in a variable gadget. We insist that any horizontal part consists of an even number of 5-squares. Thus the state of the final 5-square of the gadget of λ is opposite to that of the first 5-square. The literal gadget can be seen as a mechanical construction that transmits pressure from the variable gadget into the clause gadget: if the 5-square where a literal gadget is attached to its variable gadget from the top is in upper state (corresponding to *false*) then the active ranges of all 5-squares of the literal gadget are "pushed" towards the clause gadget; otherwise, if the literal is *true*, there is no pressure towards the clause.

For a literal λ, let m_λ be the number of regular 5-squares, and let m'_λ be the number of connectors in the gadget of λ. Then λ contributes at most $6m_\lambda + m'_\lambda$ units to H.

Clause gadgets. The final square of each literal gadget connects to a clause gadget. A clause gadget consists of two intersecting 5-squares Q_C and Q'_C, containing six 1-squares as depicted in Figure 5. If the six 1-squares contribute one unit each, the two 5-squares can also contribute at most one unit each. This is by construction also the maximum contribution of a clause gadget. Let $Q \in \{Q_C, Q'_C\}$. Note that there are three scale intervals in which Q might contribute one unit to H, and that Q intersects the final 5-square of two of the three legs corresponding to literals in C. Note that the literal gadgets have enough slack to make the final 5-squares intersect Q as shown in Figure 5. Assume that the literal legs contribute maximally to H. Then, if the two literal legs intersecting Q evaluate to *false*, only the middle unit-height strip of Q can contribute (one unit) to H. But since Q_C and Q'_C intersect, their two middle strips *together* can contribute at most one unit. Thus, if all three literals evaluate to *false*, the clause gadget (Q_C, Q'_C and the six 1-squares in their union) contributes 7 units in total. On the other hand, if at least one literal in C evaluates to *true*,

both Q_C and Q'_C can contribute one unit, and the clause gadget contributes 8 units. This is the case in Figure 5, where Q_C contributes its middle strip and Q'_C contributes its top strip.

Reduction. The variable, literal, and clause gadgets form the set $\mathcal{E}_\varphi$ of extrusions representing φ. It remains to fix the threshold K such that φ is satisfiable if and only if $H(\mathcal{S}) \geq K$. Note that a variable gadget contributes maximally to H if and only if the states of its 5-squares alternate, that is, it correctly encodes either the value *true* or the value *false*. Similarly, a literal leg contributes maximally to H if and only if it correctly transfers the value of the literal (or *false*) to its final 5-square. Finally, a clause gadget contributes maximally to H if and only if at least one of its literal legs encode the value *true*. Thus φ is satisfiable if and only if all gadgets in our construction contribute maximally to H.

Let $K = (8\sum_{v\in \mathrm{Var}(\varphi)} n_v - 2) + (\sum_{\lambda\in \mathrm{Lit}(\varphi)} 6m_\lambda + m'_\lambda) + 8m$, where $\mathrm{Var}(\varphi)$ and $\mathrm{Lit}(\varphi)$ denote the sets of variables and literals in φ, respectively. The summands of K correspond to the maximum contributions of all variable gadgets, literal gadgets, and clause gadgets. By the above observation a total active range height of at least K can be achieved if and only if φ is satisfiable.

The set $\mathcal{E}_\varphi$ consists of $O(m^2)$ squares since the variable-clause graph of φ can be drawn on a grid of size $O(m^2)$ [10]. The positions of all squares can be encoded in space quadratic in the length of an encoding of φ. The reduction can be performed in polynomial time. □

2.2. General 1d-ARO with proportional dilation

We have not succeeded in showing that general 1d-ARO is hard when all extrusions are (*i*) unit squares, (*ii*) unit-width rectangles, or (*iii*) trapezoidal segments of congruent triangles with apexes on the x-axis. By specialization, problem (*ii*) is at least as hard as problem (*i*). It turns out that problem (*iii*) is at least as hard as problem (*ii*). The idea behind the proof (see the appendix) is that an instance of (*ii*) can be scaled (in polynomial space) such that the resulting trapezoidal segments nearly look like unit-width rectangles and behave like them, too.

Theorem 2. *There is a polynomial-time reduction from general 1d-ARO with constant dilation for unit-width rectangles (problem (ii)) to general 1d-ARO with proportional dilation (problem (iii)).*

2.3. Simple 2d-ARO with proportional dilation

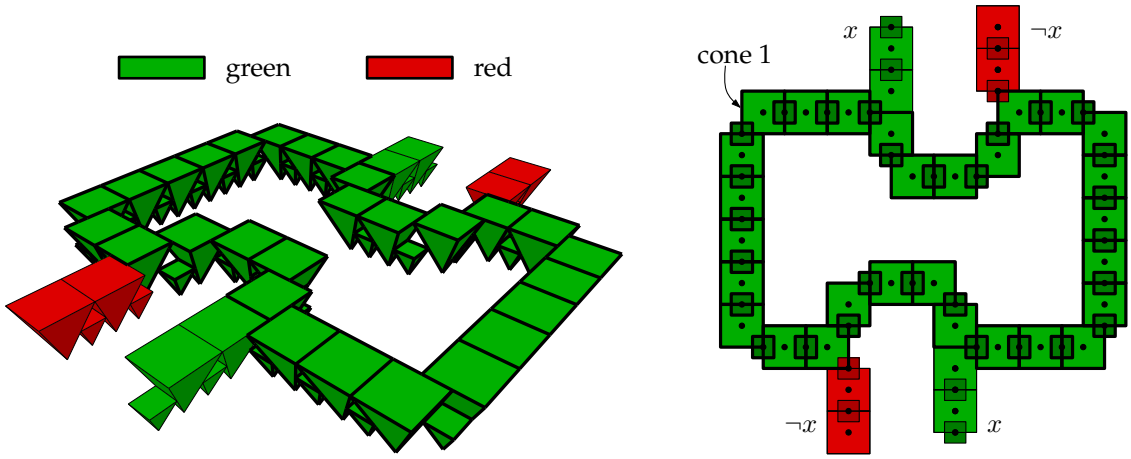
In contrast to 1d-ARO, in 2d even the *simple* ARO problem is hard.

Theorem 3. *Simple 2d-ARO with proportional dilation is NP-complete; that is, given a set $\mathcal{E}$ of axis-aligned rectangular cones and a real number $K > 0$, it is NP-complete to decide whether there is a valid assignment $\mathcal{S}$ of active ranges to $\mathcal{E}$ with $H(\mathcal{S}) \geq K$. The problem remains NP-complete when restricted to instances where all extrusions are congruent square cones.*

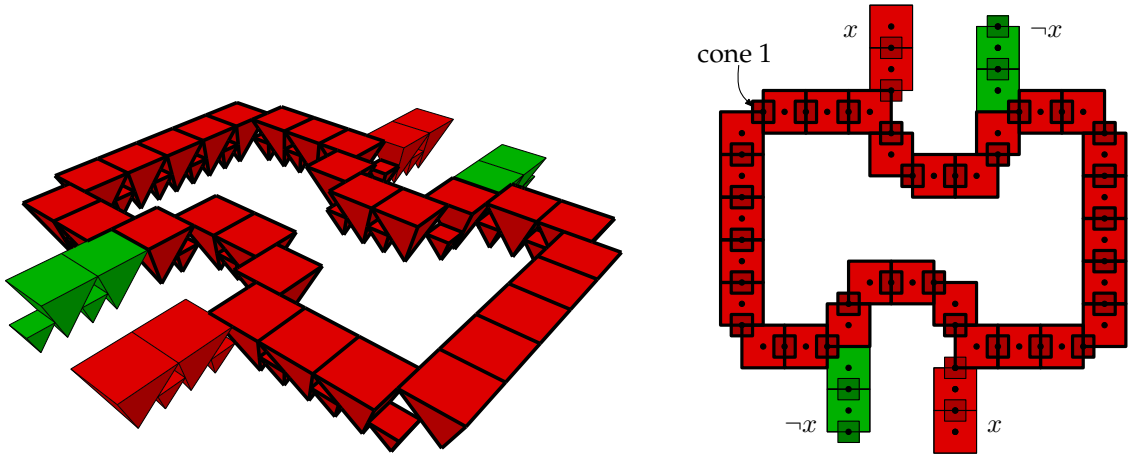
Proof. To see membership in $\mathcal{NP}$, note that in any optimal solution $\mathcal{S}$ each cone either reaches $S_{\max}$ or touches another cone. Thus $\mathcal{S}$ can be constructed by guessing an order in which the cones are greedily "filled" from bottom to top as far as possible.

To show hardness, let φ be a planar 3SAT formula with n variables and m clauses. We construct a set $\mathcal{E}_\varphi$ of congruent unit square cones and show that there is a threshold $K > 0$ such that $H(\mathcal{S}) \geq K$ for an optimal solution $\mathcal{S}$ if and only if φ is satisfiable.

Similar to the 1d-case, we construct variable gadgets and three-legged combs connecting them to clause gadgets. Figure 6 shows a variable gadget and Figure 7 shows a clause gadget. Variable gadgets and comb legs consist of chains of cones whose apexes lie on a half-integer grid, that is, a grid whose grid point coordinates are integer multiples of 1/2. Cones whose apexes have L_∞-distance 1/2 intersect at scale $S_{\max}/2$ while those with L_∞-distance 1 touch at scale $S_{\max}$. Therefore, in an optimal solution all cones are active and do not interfere in the range $(0, S_{\max}/2)$, but only every other cone in a chain can extend up to $S_{\max}$. We say that two cones are *adjacent* if their L_∞-distance is 1/2.



(a) Variable gadget for $x = \mathit{true}$.

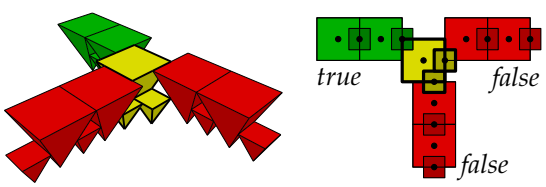


(b) Variable gadget for $x = \mathit{false}$.

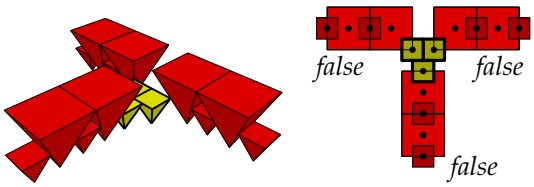
Figure 6: 3d-models and 2d-projections of a variable gadget (thick edges) with partial literal gadgets (thin edges). Cones of gadgets carrying the value *true* are green, cones of gadgets carrying the value *false* are red.

Variable gadgets. A variable gadget consists of a cyclic chain of an even number of adjacent cones. By the above observation this chain contributes maximally to H if every other cone is active in the range $(0, S_{\max})$ and the remaining cones in the range $(0, S_{\max}/2)$. Numbering the cones clockwise starting with the leftmost cone in the top row of the gadget, we denote the state where the odd cones extend to the full scale $S_{\max}$ as *true* and the state where the even cones extend to $S_{\max}$ as *false*; see Figure 6.

Literal gadgets. Each variable gadget has indentations. Their number depends on how often the variable occurs in the clauses of φ. At each indentation we can connect a leg of a three-legged comb that serves as a literal gadget. Each leg consists of an even chain of adjacent cones leading towards the clause gadget. The middle leg of a comb is a simple vertical chain, whereas the left and right leg start vertically and then bend 90 degrees in order to reach the clause gadget horizontally. For a positive literal the leg is adjacent to the beginning (in clockwise order) of the indentation, for a negative literal the leg is adjacent to the end of the indentation, see Figure 6. Thus, if a literal evaluates to *false*, the corresponding literal leg must start with a cone of height $S_{\max}/2$; otherwise it can start with a cone of height $S_{\max}$. A literal leg contributes maximally to H if every other cone reaches $S_{\max}$. Hence it has two maximal states—either the odd or the even cones reach $S_{\max}$.



(a) Clause gadget, where one literal is *true*.



(b) Clause gadget, where all literals are *false*.

(c) Color legend.

Figure 7: 3d-models and 2d-projections of a clause gadget (three yellow cones with thick edges in the center) and partial literal gadgets (thin edges). Cones of literal gadgets carrying the value *true* are green, cones of literal gadgets carrying the value *false* are red.

Clause gadgets. The clause gadget in Figure 7 consists of three pairwise adjacent cones, that is, at most one of them can extend to $S_{\max}$, and their maximal total contribution to H is $2S_{\max}$. Each of the clause cones is adjacent to the last cone of one of the three incoming literal legs. Since the literal legs consist of an even number of cones, the leg for a *true* literal ends in a cone of height $S_{\max}/2$, while a *false* literal leg ends in a cone of height $S_{\max}$—at least if the literal legs contribute maximally to H. Thus, none of the three clause cones can be of height $S_{\max}$ if and only if all literals in the clause are *false*.

We claimed above that literal gadgets always consist of an even number of cones. This ensures that in both states, they contribute the same amount to H. Now assume that a specific literal gadget actually consists of an odd number of cones. Then we replace a straight part consisting of twelve cones by the parity inverter gadget depicted in the top part of Figure 8. For comparison, the bottom part of Figure 8 shows a straight chain of cones. Note that the inverter gadget and the straight chain span the same distance on the grid; the inverter with an odd number of cones, the chain with an even number.

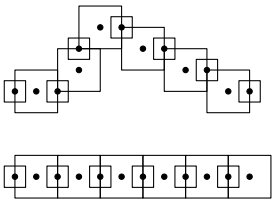


Figure 8: Parity inverter gadget (top).

Reduction. As before we determine the threshold K such that φ is satisfiable if and only if $H(\mathcal{S}) \geq K$ and all variable, literal, and clause gadgets contribute maximally to H. Variable and literal gadgets contribute maximally if they correctly encode their truth values, and clause gadgets contribute maximally if at least one of their literals evaluates to true and thus if the clause is satisfied.

We denote the number of cones for the variable gadget of variable x by n_x and the number of cones for the literal leg of literal λ by m_λ. These numbers depend on φ and are fixed in the construction. Let

$$K = 3/4 \cdot S_{\max} \sum_{v \in \mathrm{Var}(\varphi)} n_v + 3/4 \cdot S_{\max} \sum_{\lambda \in \mathrm{Lit}(\varphi)} m_\lambda + 2m \cdot S_{\max},$$

where $\mathrm{Var}(\varphi)$ and $\mathrm{Lit}(\varphi)$ denote the variables and literals in φ, respectively. The summands of K correspond to the maximum contributions of all variable gadgets, literal gadgets, and clause gadgets. By the above

observation on maximal contributions, a total active range height $H(\mathcal{S})$ of at least K can be achieved by an optimal solution $\mathcal{S}$ if and only if φ is satisfiable.

As in the proof of Theorem 1, the set $\mathcal{E}_\varphi$ consists of $O(m^2)$ extrusions placed on a grid of size $O(m^2)$, and the reduction takes polynomial time. □

3. Algorithmic toolbox

We give a toolbox of six different algorithms to tackle several variants of 1d- and 2d-ARO problems, using dynamic programming, a left-to-right greedy algorithm, line stabbing, divide and conquer, a top-to-bottom fill-down sweep, and a level-based small-to-large greedy algorithm.

3.1. Dynamic programming

Triangles. We start by considering simple 1d-ARO with proportional dilation: each extrusion E is a triangle with apex on the x-axis and top edge on the horizontal line $s = S_{\max}$. The truncated extrusions differ only by having (possibly) lower top edges. Observe that in an optimal solution at least one truncated extrusion has height $S_{\max}$, and thus divides the problem into two independent subproblems. This is the essence of our dynamic program. Details can be found in the appendix.

Theorem 4. *Simple 1d-ARO with proportional dilation can be solved in $O(n^3)$ time using $O(n^2)$ space.*

3.2. Left-to-right greedy algorithm

Unit-height rectangles. Van Kreveld et al. [16] presented the following greedy algorithm for maximum independent set (MIS) among axis-aligned rectangles of unit height. We are given a set $\mathcal{E}$ of unit-height rectangles. Until $\mathcal{E}$ is empty, repeatedly select the rectangle $E \in \mathcal{E}$ with leftmost right edge, and remove from $\mathcal{E}$ all rectangles intersecting E. This takes $O(n \log n)$ time, and is a $(1/2)$-approximation for MIS [16]. It is not hard to see that the same algorithm yields a $(1/3)$-approximation for 1d-ARO.

Theorem 5. *The maximum total active range height for a set of n rectangular extrusions of unit height can be approximated within a factor of $1/3$ in $O(n \log n)$ time.*

3.3. Line stabbing

We use line stabbing for unit squares and unit-width rectangles, that is, general 1d-ARO with constant dilation and equal-size labels. Line stabbing is a special case of the shifting technique by Hochbaum and Maass [9]. Agarwal et al. [1] have used line stabbing to design a PTAS for maximum independent set among unit-width rectangles. The idea is to stab the extrusions with vertical lines such that two lines have distance at least 1, each extrusion is stabbed by exactly one line, and each line stabs at least one extrusion. Such a set of lines can be computed greedily in linear time if the extrusions are given in left-to-right order. We assume that the resulting lines are numbered l_1 to l_k from left to right, where $k \leq n$. Since all extrusions are open and have unit width, those intersecting line l_i for some $1 \leq i \leq k-2$ are disjoint from those intersecting line l_{i+2}. The problem for a single stabbing line can be solved optimally by a simple greedy algorithm, see the appendix.

Lemma 1. *The maximum total active range height of a set $\mathcal{E}$ of n rectangles stabbed by a vertical line can be computed in $O(n \log n)$ time.*

Unit-width rectangles. For unit-width rectangles we partition the vertical stabbing lines into sets Λ_1 and Λ_2, containing all the stabbing lines with odd and even indices, respectively. By Lemma 1 the solution for each individual stabbing line, and thus also the solution $\mathcal{A}_i$ for all rectangles intersecting lines in Λ_i, can be computed optimally for $i \in \{1, 2\}$. From the candidate solutions $\mathcal{A}_1$ and $\mathcal{A}_2$ we choose the one maximizing H as our approximate solution $\mathcal{A}$. Re-using arguments of Agarwal et al. [1], it is easy to see that $\mathcal{A}$ is a $(1/2)$-approximation. Now Lemma 1 yields the following.

Theorem 6. *The maximum total active range height for a set of n rectangular extrusions of unit width can be approximated within a factor of $1/2$ in $O(n \log n)$ time.*

Unit squares. For unit squares, we partition the vertical stabbing lines into *three* sets, $\Lambda_i = \{l_j \mid j = i \bmod 3\}$ for $i = 1, 2, 3$. Deleting all squares stabbed by one of the sets Λ_1, Λ_2, or Λ_3 divides the problem into independent subproblems defined by two consecutive stabbing lines each. A greedy sweep-line algorithm finds the optimal solution for each of these subproblems as follows.

Lemma 2. *The maximum total active range height of a set $\mathcal{E}$ of n unit squares stabbed by two vertical lines of distance at least 1 can be computed in $O(n \log n)$ time.*

Proof. The algorithm sweeps a line from top to bottom over $\mathcal{E}$, greedily activating the maximum number of independent squares at each scale. Specifically, we greedily activate the leftmost square on the left stabbing line and the rightmost square on the right stabbing line, if they are disjoint. If they intersect then preference is given to whichever was activated first.

Clearly the activated portions are pairwise disjoint and the sum of their heights is no less than optimal. We now show that each square is activated over at most one range. Once a square E is activated it can be deactivated before the end of its selectable range only if it is superseded by a more extremal square E' (farther left on the left line or farther right on the right line). This can happen only at scale $S_{E'} < S_E$. Thus, E' will continue to be more extremal through scale s_E, and E can never be reactivated.

The running time is obvious. □

Using the algorithm described above, we optimally solve the three subproblems created by removing the squares stabbed by, respectively, one of the sets Λ_1, Λ_2, and Λ_3. Again, using arguments of Agarwal et al. [1], one easily sees that the sub-solution that maximizes H is a (2/3)-approximation.

Theorem 7. *The maximum total active range height for a set of n unit-square extrusions can be approximated within a factor of 2/3 in $O(n \log n)$ time.*

3.4. Divide and conquer

Arbitrary rectangles. Agarwal et al. [1] gave an $O(n \log n)$-time divide-and-conquer algorithm to compute a $(1/\log n)$-approximation for MIS among axis-aligned rectangles. Their algorithm readily adapts to ARO. In both cases the one-dimensional subproblems can be solved optimally (see Lemma 1), which yields the base of the inductive proof of the approximation factor.

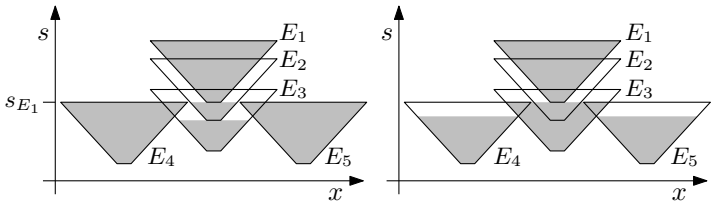
Theorem 8. *The divide-and-conquer algorithm computes in $O(n \log n)$ time a $(1/\log n)$-approximation to the maximum total active range height for a set of n rectangles.*

3.5. Top-to-bottom fill-down sweep

In this subsection we introduce a simple sweep-line algorithm that yields constant-factor approximations for several variants of 1d- and 2d-ARO. The idea is to sweep a line or plane downwards over the extrusions in $\mathcal{E}$, and when reaching a scale s at which $E \in \mathcal{E}$ can be selected without conflicting with any previously selected extrusion, we "fill" E from s down to its bottom—that is, we set $(a_E, A_E) = (s_E, s)$. Once the active range of an extrusion E is set, we call E *fixed*. Whenever we have the choice of selecting one of two intersecting extrusions, we select the one whose selectable range has the higher upper end point. Figure 9 shows the importance of breaking ties in the correct order. This order ensures that we can derive the approximation factors in Lemma 5 and Theorem 11.

Note that with this algorithm we have $a_E = s_E$ for every E that contributes to the objective function H. This corresponds to the effect that during zoom-in a label only disappears when its selectable range ends. This is likely to be desirable in applications.

For the pseudo-code of this algorithm—see Algorithm 1—we need one new piece of notation. Refer to Figure 10. For each pair of distinct extrusions $E, E' \in \mathcal{E}$, let $s_{EE'} \leq \min\{S_E, S_{E'}\}$ be the largest scale such that $E \cap E' \cap \pi = \emptyset$, where π is a horizontal plane at $s = s_{EE'}$. Due to the types of dilation we consider, we know that $E \cap E' \cap \pi' \neq \emptyset$ for any horizontal plane π' strictly between π and the horizontal plane at $s = \min\{S_E, S_{E'}\}$. For the types of dilation we consider, we can compute $s_{EE'}$ in constant time.



(a) E_2 is correctly selected first (b) E_3 is selected first

Figure 9: It matters in which order extrusions E_2 and E_3 are considered when the algorithm reaches scale s_{E_1}. (At that point we have $A_{E_2} = A_{E_3}$ and $S_{E_2} > S_{E_3}$.) The total active range height in case (b) is less than in case (a).

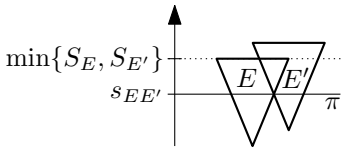


Figure 10: For a pair of distinct extrusions $E, E' \in \mathcal{E}$, $s_{EE'} \leq \min\{S_E, S_{E'}\}$ is the largest scale such that $E \cap E' \cap \pi = \emptyset$, where π is a horizontal plane at $s = s_{EE'}$.

Algorithm 1: Top-to-bottom fill-down sweep

Input : a set $\mathcal{E}$ of extrusions, a selectable range (s_E, S_E) for each $E \in \mathcal{E}$
Output: an active range (a_E, A_E) for each $E \in \mathcal{E}$

```
foreach E ∈ E do (a_E, A_E) ← (s_E, S_E)
Q ← priority queue for E sorted lexicographically by (A_E, S_E)
while Q ≠ ∅ do
    remove first element E from Q and fix its current active range
    foreach E' ∈ Q do
        if E' ∩ E ≠ ∅ then
            A_E' ← min(A_E', s_EE')
            if A_E' = a_E' then remove E' from Q
```

The correctness of Algorithm 1 is clear: each extrusion is assigned at most one active range that is a subset of its selectable range, and none of the truncated extrusions intersect.

We now describe a generic implementation, that is, one that makes no further assumption about the shapes of the extrusions. Our algorithm makes at most n steps. In each step, we make two passes through the extrusions that are not fixed yet. In the first pass, we select the extrusion E with the lexicographically largest pair (A_E, S_E). In the second pass, we adjust the upper end points of the active ranges of the others. This immediately gives the following result.

Lemma 3. *A generic implementation of Algorithm 1 runs in $O(n^2)$ time and uses $O(n)$ space.*

Let $\mathcal{A} = \{(a_E, A_E) \mid E \in \mathcal{E}\}$ be the solution computed by Algorithm 1. Recall that the trace $\mathrm{tr}_{s^\star}(E)$ of an extrusion E at scale $s^\star$ is the intersection of E with the horizontal line or plane $s = s^\star$ in 1d- and 2d-ARO, respectively. We say that an extrusion E *blocks* another extrusion E' at scale s under a given solution if the traces of E and E' at s intersect and $s \in (a_E, A_E)$. Note that this implies that $s \notin (a_{E'}, A_{E'})$. We say that two extrusions are *independent at s* if their traces at scale s are disjoint. The following lemma will help prove all approximation factors in this section.

Lemma 4. *If, for any set $\mathcal{E}$ of extrusions, for any $E \in \mathcal{E}$, and for any scale $s \geq 0$, E blocks no more than c pairwise independent extrusions at s, then Algorithm 1 computes a $(1/c)$-approximation for the maximum total active range height.*

Proof. In Algorithm 1, each extrusion E starts with $A_E = S_E$, and A_E is lowered only as far as is necessary to avoid conflicts with fixed extrusions. Thus, if $E \in \mathcal{E}$ is inactive in $\mathcal{A}$ at scale s, then E must be blocked at s.

If at any scale s no extrusion can block more than c mutually independent extrusions, and in $\mathcal{A}$ every extrusion that is inactive at s is blocked at s, then the number of extrusions that are active in an optimal solution at scale s can be no more than c times the number in $\mathcal{A}$. Integrating over all scales proves the lemma. □

For each of the extrusion shapes covered in this section we determine a value for c, usually 2 or 4. For example, it is easy to see that $c = 2$ holds for unit-width rectangles (or, more generally, for any set of rectangles where the x-order of the left edges is the same as the x-order of the right edges), so Algorithm 1 yields a (1/2)-approximation. (Theorem 6 states the same result via a different algorithm based on line stabbing.)

In the remainder of this subsection we discuss different extrusion types. For each type we analyze the performance of Algorithm 1 in terms of approximation factor, as well as time and space complexity, which can in most cases be improved over the generic implementation.

Congruent trapezoids. In this part we consider the 1d-ARO problem for congruent trapezoids as the extrusion shapes.

Lemma 5. *Algorithm 1 approximates the maximum total active range height of a set of n congruent trapezoids within a factor of $1/2$.*

Proof. Note first that the trace of each trapezoid at any scale s is a line segment. We claim that if a trapezoid E blocks another trapezoid E' at scale s under solution $\mathcal{A}$, then $\mathrm{tr}_s(E')$ must contain at least one endpoint of $\mathrm{tr}_s(E)$. This immediately implies $c = 2$ in Lemma 4, and thus yields the approximation factor of 1/2.

We now prove our claim above. Suppose that under $\mathcal{A}$, E blocks E' at s, meaning that their traces at s intersect and that $A_{E'} < s \leq A_E$. If $S_{E'} \leq S_E$, then, since E and E' are congruent, $\mathrm{tr}_s(E')$ is at least as wide as $\mathrm{tr}_s(E)$ and clearly $\mathrm{tr}_s(E')$ must contain at least one endpoint of $\mathrm{tr}_s(E)$, see Figure 9a using $E = E_2$ and $E' = E_3$. If $S_{E'} > S_E$, then, due to the order in which Algorithm 1 selects extrusions, $A_{E'}$ must have been reduced below A_E before E was selected. This means that E' is blocked by some other extrusion E'' at scale A_E. Since E is not blocked by E'' at A_E, $\mathrm{tr}_{A_E}(E')$ is not completely contained in $\mathrm{tr}_{A_E}(E)$. Thus $\mathrm{tr}_{A_E}(E')$ must contain at least one endpoint of $\mathrm{tr}_{A_E}(E)$ as they have to intersect. Finally, since the two trapezoids E and E' are congruent and their traces at s still intersect, $\mathrm{tr}_s(E')$ must still contain at least one endpoint of $\mathrm{tr}_s(E)$. □

The order of breaking ties when selecting extrusions in Algorithm 1 matters, see the example in Figure 9. Figure 9a shows the output of Algorithm 1 based on the fact that at scale s_{E_1}, extrusion E_2 has higher priority than E_3 (since $A_{E_2} = A_{E_3}$ and $S_{E_2} > S_{E_3}$). Figure 9b shows that switching this order would allow one extrusion (E_3) to block *three* independent extrusions (E_2, E_4, and E_5).

For congruent trapezoids we can improve on the generic implementation by using more specific data structures. We build upon the standard algorithm for finding intersections among a set of line segments [6, Theorem 2.4]. That algorithm stores a set of line segments intersecting a horizontal sweep line in a binary search tree ordered horizontally, and considers only intersections between neighboring segments.

We run this algorithm in parallel with maintaining our priority queue $\mathcal{Q}$. In our case, the line segments are the side edges of the fixed trapezoids. We replace the body of the main loop in Algorithm 1 with the following. Remove the top-priority extrusion E from $\mathcal{Q}$ and check (in $O(\log n)$ time) whether it intersects any fixed extrusion. If not, fix E by fixing its active range and insert the side edges of E into the binary search tree. Otherwise, reduce A_E to $\min_{E' \in \mathcal{E}'} s_{EE'}$, where $\mathcal{E}'$ contains the (at most two) fixed extrusions intersecting E, and put E back into $\mathcal{Q}$ with its new priority (A_E, S_E).

In order to bound the running time of this improved algorithm, note that we can attribute each time we put the current extrusion back into $\mathcal{Q}$ to a unique pair of trapezoid side edges that intersect. Together with Lemma 5 this yields the following.

Theorem 9. *A (1/2)-approximation for the maximum total active range height of a set of n congruent trapezoids can be computed in $O((k + n) \log n)$ time and $O(n)$ space, where k is the number of side-edge intersections between pairs of trapezoids.*

Trapezoidal segments of congruent triangles. Here we consider the 1d-ARO problem with proportional dilation, where the extrusions are trapezoidal segments of congruent underlying triangles that have their apexes on the x-axis.

Theorem 10. *A $(1/2)$-approximation for the maximum total active range height of a set of n trapezoidal segments of congruent triangles can be computed in $O((k+n)\log n)$ time and $O(n)$ space, where k is the number of side-edge intersections between pairs of trapezoids.*

Proof. Since the underlying triangles are congruent and horizontally aligned, the width of every trapezoid is the same at each scale. This implies that any trapezoid blocked by another trapezoid E intersects a side edge of E. Thus, at most two extrusions blocked by E at scale s can be independent at s, and the approximation factor $1/2$ follows from Lemma 4.

We can use the same implementation as for congruent trapezoids. □

Congruent frusta. Axis-aligned congruent square frusta are the 2d-ARO analogues of congruent trapezoids. Here, a blocked frustum must intersect a side *face* of its blocker. The number of independent frusta that can intersect a single face depends on W, the ratio of the length of the top edges of each frustum to the length of their bottom edges.

Theorem 11. *A $1/(4W)$-approximation for the maximum total active range height of a set of n axis-aligned congruent square frusta can be computed in $O(n^2)$ time and $O(n)$ space, where W is the ratio of the length of the top edges of each frustum to the length of its bottom edges.*

Proof. Using an argument similar to that used in the proof of Lemma 5, it can be shown that if a frustum E blocks another frustum E' at scale s under solution $\mathcal{A}$, then $\mathrm{tr}_s(E')$ must intersect the *boundary* of $\mathrm{tr}_s(E)$. Therefore, since at any scale s the side length of a trace of E is no more than W times the side length of a trace of E', E can block at most $4W$ independent extrusions at s. For an example with $W = 3$, see Figure 12. The approximation factor of $1/(4W)$ follows from Lemma 4. Running time and space requirements follow directly from Lemma 3. □

Frustal segments of congruent square cones. In this part we are concerned with extrusions that are frustal segments of underlying axis-aligned congruent square cones with their apexes at $s = 0$. This is the 2d-ARO equivalent of trapezoidal segments of congruent triangles, which we have treated in Theorem 10. As in the case of trapezoids, we want to efficiently compute the set of (active) fixed extrusions that intersect a given non-fixed extrusion. Now, however, the line-intersection approach is not sufficient. Instead, we represent each extrusion by the position of the apex of its underlying cone. Using this representation, we maintain the set of active fixed extrusions in a range tree [6, Chapter 5.3].

Theorem 12. *A $(1/4)$-approximation for the maximum total active range height of a set of n frustal segments of axis-aligned congruent square cones can be computed in $O((k+n)\log^2 n)$ time and $O(n\log n)$ space, where k is the number of side-face intersections between pairs of frusta.*

Proof. Since the trace of every extrusion at some scale s has the same constant size, we know that any extrusion blocked by an extrusion E at s must intersect one of the four corner edges of E at s. (Two side faces meet at a corner edge.) This means that at most four blocked extrusions can be independent and the approximation factor of $1/4$ follows from Lemma 4.

The implementation is the same as for congruent trapezoids, except that the range tree requires $O(n\log n)$ space and that updates and queries in a range tree take $O(\log^2 n)$ time. The query time is due to the fact that every query returns $O(1)$ results. □

Simple ARO with axis-aligned congruent square cones is a special case of the above, where $(s_E, S_E) = (0, S_{\max})$ for each $E \in \mathcal{E}$, so we immediately get the following corollary.

Corollary 1. *A $(1/4)$-approximation for the maximum total active range height of a set of n axis-aligned congruent square cones can be computed in $O((k+n)\log^2 n)$ time and $O(n\log n)$ space, where k is the number of pairs of intersecting cones.*

Recall that the generic implementation requires $O(n^2)$ time and $O(n)$ space, which might be preferable to the above implementation, depending on the size of k and on the relative importance of space and time.

3.6. Level-based small-to-large greedy algorithm

In this section we give an algorithm for simple 2d-ARO with cones whose bases are axis-aligned squares. Rather than sweeping the events defined by the extrusions themselves, the algorithm in this section is based on intersecting the cones with $O(\log n)$ horizontal planes or *levels*. On each of these levels we activate some extrusions before we proceed to the next (lower) level. This yields a (1/24)-approximation for arbitrary cones and a $(1/4 - \varepsilon)$-approximation for congruent cones. The latter result is slightly worse than the 1/4-approximation stated in Corollary 1, but the running time of the level-based algorithm is independent of k, the number of pairs of intersecting cones, which may be quadratic in n. We start with the general result.

Arbitrary axis-aligned square cones. Note that for any extrusion E and any scale s, the trace $\mathrm{tr}_s(E)$ is a square. We call $\mathrm{tr}_s(E)$ *(in-)active* if E is (in-)active at scale s. Analogously, we call $\mathrm{tr}_s(E)$ *blocked* if E is blocked at s. Since we consider simple ARO, it holds that for each $E \in \mathcal{E}$, $a_E = 0$, and it remains to set A_E.

See Algorithm 2 for the pseudo-code of our algorithm. It works in $O(\log n)$ phases, where phase i deals with the situation on a horizontal plane π_i at scale $s_i = S_{\max}/\alpha^i$. The phases are numbered 0 to $N_k = \lceil \log_\alpha kn \rceil$, where $\alpha = (k+1)/k$ and $k \geq 1$ is an integer-valued parameter that will become important only when we treat congruent square cones below. Here we set $k = 1$, which yields $\alpha = 2$ and $N_1 = \lceil \log_2 n \rceil$. For ease of presentation we add a dummy plane π_{N_k+1} at scale $s = 0$.

Algorithm 2: Level-based algorithm for 2d-ARO

Input : set of square cones $\mathcal{E}$, each with selectable range $(0, S_{\max})$
Output: active range $(0, A_E)$ for each $E \in \mathcal{E}$

foreach $E \in \mathcal{E}$ **do** initialize E as inactive and set $A_E \leftarrow 0$
for $i = 0$ **to** N_k **do**
  { phase i }
  $s_i \leftarrow S_{\max}/\alpha^i$
  $\mathcal{C}_i \leftarrow$ set of traces that are inactive at s_i and that intersect no traces active at s_i
  while $\mathcal{C}_i \neq \emptyset$ **do**
    $T \leftarrow$ smallest trace in $\mathcal{C}_i$
    mark T as active and set $A_E \leftarrow s_i$, where E is the extrusion of T
    remove T and all traces intersecting it from $\mathcal{C}_i$

Let $0 \leq i \leq N_k$. When the algorithm finishes phase i, all traces that are inactive at level i intersect an active trace—they are blocked. Let $s = s_i$. For analyzing the performance of our algorithm, we consider level i and associate each blocked trace T to one of the active traces in the following way:

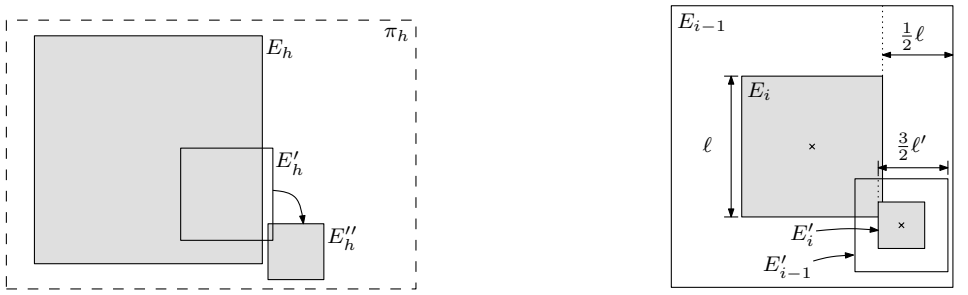
(i) if T was not blocked at the beginning of phase i but became blocked by the trace $T^\star$ of a newly activated extrusion, then associate T to $T^\star$;

(ii) if T was blocked in the beginning of phase i, then associate T to any trace that is blocking T and that was active at the beginning of phase i.

Next, we show that the traces associated to an active trace cannot be arbitrarily small.

Lemma 6. *Let $0 \leq i \leq N_k$ and let T be a trace active at level i with side length ℓ. Then any trace associated to T has side length at least $\ell/3$ and intersects the boundary of T.*

Proof. For an extrusion E and a level i we denote the trace $\mathrm{tr}_{s_i}(E)$ in short by E_i. Then for the given trace T we have $T = E_i$ for some extrusion E.



(a) Trace E'_h is associated to E''_h.

(b) Trace E_{i-1} contains E'_{i-1}.

Figure 11: Intersection behavior of the traces of E, E', and E'' at different levels.

Let T' be a trace associated to T. If T' is associated to T by case (i) above we know that the side length of T' is at least ℓ by the order in which Algorithm 2 selects the squares.

So assume $T' = E'_i$ is associated to $T = E_i$ by case (ii) and let $h < i$ be the largest level in which E'_h is *not* associated to E_h. (If there is no such level h then case (i) applies to E'_0 and E_0 and the statement of the lemma holds.) Thus at level h we have E'_h associated to some other active trace E''_h. Since E_{h+1} blocks E'_{h+1} in the beginning of phase $h+1$ we know that E_h is already active and hence does not intersect E''_h. On the other hand, both E_h and E''_h must intersect E'_h. This situation is depicted in Figure 11a.

Let ℓ' be the side length of T' and suppose $\ell' < \ell/3$. Now consider level $i-1$. There, the side lengths of the traces E'_{i-1} and E_{i-1} are doubled, which means that E'_{i-1} is fully contained in E_{i-1}, see Figure 11b. This also holds for level $h \leq i-1$ and thus E'_h cannot intersect the active trace E''_h, since E''_h is disjoint from E_h—a contradiction. For the same reason T' must intersect the boundary of T. □

We now fix an optimal solution $\mathcal{S}$. Let $\mathcal{A}$ denote the solution of Algorithm 2. For $i = 1, \ldots, N_k + 1$ we denote the active segments of the extrusions between planes π_{i-1} and π_i in $\mathcal{S}$ by $\mathcal{S}_i$ and in $\mathcal{A}$ by $\mathcal{A}_i$, respectively. We charge $H(\mathcal{S}_{N_k})$ and $H(\mathcal{S}_{N_k+1})$ to $H(\mathcal{A}_1)$, and for $1 \leq i < N_k$ we charge $H(\mathcal{S}_i)$ to $H(\mathcal{A}_{i+1})$.

Lemma 7. *It holds that $H(\mathcal{A}_1) \geq (H(\mathcal{S}_{N_k}) + H(\mathcal{S}_{N_k+1}))/\alpha^2$, and if no more than c traces of extrusions in $\mathcal{S}$ are associated to any given trace in $\mathcal{A}$, then for $i = 1, 2, \ldots, N_k - 1$ it holds that $H(\mathcal{A}_{i+1}) \geq H(\mathcal{S}_i)/(c\alpha)$.*

Proof. We first compare $H(\mathcal{S}_{N_k}) + H(\mathcal{S}_{N_k+1})$ to $H(\mathcal{A}_1)$. The scale of π_{N_k-1} is at most $\alpha \cdot S_{\max}/(kn)$ and obviously there are at most n active cone segments in $\mathcal{S}$ below π_{N_k-1}, so their total active range height is at most $\alpha/k \cdot S_{\max}$. On the other hand, there is at least one active cone segment in $\mathcal{A}_1$ of height $S_{\max}/(k+1)$. Using $\alpha = (k+1)/k$ yields $H(\mathcal{A}_1) \geq (H(\mathcal{S}_{N_k}) + H(\mathcal{S}_{N_k+1}))/\alpha^2$.

Now let $E \in \mathcal{E}$ and let $1 \leq i < N_k$. If $\mathrm{tr}_{s_i}(E)$ is active in $\mathcal{S}$, either $\mathrm{tr}_{s_i}(E)$ is active also in $\mathcal{A}$ or $\mathrm{tr}_{s_i}(E)$ is associated to a trace $T' = \mathrm{tr}_{s_i}(E')$ that is active in $\mathcal{A}$.

In the first case, E contributes to $H(\mathcal{A}_{i+1})$ at least $1/\alpha$ times what it contributes to $H(\mathcal{S}_i)$ since the height difference between π_{i-1} and π_i is α times the height difference between π_i and π_{i+1}.

In the second case, let $\mathcal{S}(T')$ be the set of extrusions in $\mathcal{S}$ whose traces are associated to T'. By assumption we have $|\mathcal{S}(T')| \leq c$. Since we must also take into account the ratio of the height differences, we get that the contribution of E' to $H(\mathcal{A}_{i+1})$ is at least $1/(c\alpha)$ times the contribution of $\mathcal{S}(\mathrm{tr}_{s_i}(E'))$ to $H(\mathcal{S}_i)$. Summing up over all extrusions yields the statement. □

Theorem 13. *Algorithm 2 computes a $(1/24)$-approximation for the maximum total active range height of a set of n axis-aligned arbitrary square cones in $O(n \log^3 n)$ time and $O(n \log n)$ space.*

Proof. In order to analyze the performance of Algorithm 2, we consider a level i with $1 \leq i < N_k$ and an extrusion $E \in \mathcal{E}$ that is active at s_i in $\mathcal{A}$, that is, the trace $\mathrm{tr}_{s_i}(E)$ is active. Let ℓ be the side length of $\mathrm{tr}_{s_i}(E)$. By Lemma 6, all traces associated to $\mathrm{tr}_{s_i}(E)$ have side length at least $\ell/3$ and intersect the

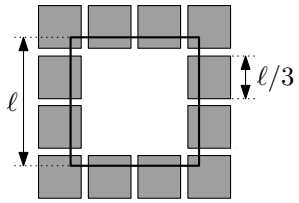


Figure 12: The boundary of a square of side length ℓ is intersected by at most twelve pairwise independent squares of side length $\ell/3$.

boundary of $\mathrm{tr}_{s_i}(E)$. Thus, at most twelve such traces can be independent in π_i and hence active in $\mathcal{S}_i$, see Figure 12. Setting $c = 12$ and $\alpha = 2$ in Lemma 7 and summing up over all levels yields $H(\mathcal{A}) \geq H(\mathcal{S})/24$.

For an efficient implementation of Algorithm 2 we store the traces (which are in fact squares) in each level i in a two-dimensional segment tree τ_i, which supports deletion in $O(\log^2 n)$ time [6, Chapter 10.3]. Let $0 \leq i \leq N_k$. For each extrusion E that has been activated at a level less than i we delete all traces intersecting $\mathrm{tr}_{s_i}(E)$ as follows. We place a vertex at each corner of $\mathrm{tr}_{s_i}(E)$, plus two more evenly spaced vertices on each edge. We then query τ_i with each of these 12 vertices and delete the returned traces from τ_i. As the side length of intersecting traces is at least $\ell/3$ (see Lemma 6) these points suffice to find all traces intersecting $\mathrm{tr}_{s_i}(E)$. From the remaining traces we iteratively choose the smallest one. By querying τ_i with the four corner points of the chosen trace, we identify and remove all intersecting traces, which are larger and thus must contain one of the corner points. Since a deletion takes $O(\log^2 n)$ time, Algorithm 2 takes $O(n \log^2 n)$ time per level. The space consumption of τ_i is $O(n \log n)$.

Since there are $O(\log n)$ levels, the total running time is $O(n \log^3 n)$. □

Congruent axis-aligned square cones. For axis-aligned congruent square cones, all traces in any horizontal plane have the same size. This has two consequences. First, it simplifies implementing Algorithm 2: any trace is "smallest." Second, at most four independent traces can intersect a given one. Thus the analysis in the previous section immediately yields a (1/8)-approximation. We can however, do better in this case by using a denser set of planes, that is, by setting the parameter k to a value larger than 1.

Recall that $\alpha = (k+1)/k$ and that π_i denotes the horizontal plane at scale $s_i = S_{\max}/\alpha^i$ for $i = 0, \ldots, N_k$, where $N_k = \lceil \log_\alpha kn \rceil$. As before, $\mathcal{S}_i$ and $\mathcal{A}_i$ denote the active segments of the extrusions between π_{i-1} and π_i in the optimal solution and the solution of our algorithm, respectively.

Theorem 14. *Algorithm 2 computes a $(1/4 - \varepsilon)$-approximation for the maximum total active range height of a set of n axis-aligned congruent square cones in $O(n \log n \cdot \log(n/\varepsilon)/\varepsilon)$ time and $O(n \log n)$ space.*

Proof. Let $\mathcal{E}$ be the given set of extrusions. In order to analyze the performance of Algorithm 2, we consider a level i with $1 \leq i < N_k$ and an extrusion $E \in \mathcal{E}$ that is active at s_i in $\mathcal{A}$, that is, $\mathrm{tr}_{s_i}(E)$ is active. Since all traces in π_i are congruent squares, there are at most four traces intersecting $\mathrm{tr}_{s_i}(E)$ that are active in $\mathcal{S}$. Setting $c = 4$ in Lemma 7 and summing up over all levels yields $H(\mathcal{A}) \geq H(\mathcal{S})/(4\alpha)$. With $k = 1/(4\varepsilon) - 1$, we get $\alpha = 1/(1 - 4\varepsilon)$ and thus a $(1/4 - \varepsilon)$-approximation.

Our implementation of Algorithm 2 uses an augmented dynamic range tree [12] for each level i. It stores, for each $E \in \mathcal{E}$, the trace $\mathrm{tr}_{s_i}(E)$ represented by the apex of E in the (x, y)-plane. This is possible since all cones are congruent and thus the mapping between apexes and traces is bijective. A query for all traces that intersect a given trace $\mathrm{tr}_{s_i}(E)$ easily translates into a range query for the corresponding apexes in the range tree. The augmented dynamic range tree can be constructed in $O(n \log n)$ time and space since only deletions and no insertions need to be supported. Each deletion takes $O(\log n)$ time; hence the running time per level is $O(n \log n)$ (see the proof of Theorem 13). The number of levels in Algorithm 2 is $O(\log_\alpha kn) = O((\log kn)/\log \alpha)$. For $k \geq 1$ we have $\alpha^k \geq 2$ and thus $1/\log \alpha \leq k$. Now we can bound the number of levels by $O(k \log(kn)) = O(\log(n/\varepsilon)/\varepsilon)$. This yields the desired running time of $O(n \log n \cdot \log(n/\varepsilon)/\varepsilon)$. The space consumption remains $O(n \log n)$. □

4. Open problems

ARO is an exciting new problem inspired by interactive web-based mapping applications, and this is the first paper with an extensive, rigorous algorithmic study. We have described a number of approximation algorithms, see Table 1 for an overview. An obvious question is whether any approximation factor can be improved, or whether any of the problems admits a polynomial-time approximation scheme. Furthermore, the complexity of general 1d-ARO is still unknown for regular shapes such as unit squares, congruent trapezoids, and segments of congruent triangles.

Mapping applications in practice often need to work with different label models, such as labels of different lengths and fonts; non-axis-aligned labels; non-rectangular labels, such as a road label that follows a curvy road; and sliding labels—that is, non-invariant point placements. Any of these raises a number of interesting theoretical questions.

Acknowledgments. We thank the anonymous referees for their valuable comments. They helped us in particular to simplify and speed up Algorithm 1.

References

[1] P. K. Agarwal, M. van Kreveld, and S. Suri. Label placement by maximum independent set in rectangles. *Comput. Geom. Theory Appl.*, 11:209–218, 1998.

[2] K. Been, E. Daiches, and C. Yap. Dynamic map labeling. *IEEE Trans. Visualization & Comput. Graphics*, 12(5):773–780, 2006.

[3] A. L. Buchsbaum, H. Karloff, C. Kenyon, N. Reingold, and M. Thorup. OPT versus LOAD in dynamic storage allocation. *SIAM J. Comput.*, 33(3):632–646, 2004.

[4] B. Chazelle and 36 co-authors. The computational geometry impact task force report. In B. Chazelle, J. E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, volume 223, pages 407–463. American Mathematical Society, Providence, RI, 1999.

[5] J. Christensen, J. Marks, and S. Shieber. An empirical study of algorithms for point-feature label placement. *ACM Trans. Graphics*, 14(3):203–232, 1995.

[6] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Berlin, third edition, 2008.

[7] T. Erlebach, K. Jansen, and E. Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM J. Comput.*, 34(6):1302–1323, 2005.

[8] M. Formann and F. Wagner. A packing problem with applications to lettering of maps. In *Proc. 7th Annu. ACM Sympos. Comput. Geom. (SoCG'91)*, pages 281–288, 1991.

[9] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 32:130–136, 1985.

[10] D. E. Knuth and A. Raghunathan. The problem of compatible representatives. *SIAM J. Discr. Math.*, 5(3):422–427, 1992.

[11] D. Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11(2):329–343, 1982.

[12] K. Mehlhorn and S. Näher. Dynamic fractional cascading. *Algorithmica*, 5:215–241, 1990.

[13] I. Petzold, G. Gröger, and L. Plümer. Fast screen map labeling—data-structures and algorithms. In *Proc. 23rd Internat. Cartographic Conf. (ICC'03)*, pages 288–298, Durban, South Africa, 2003.

[14] I. Petzold, L. Plümer, and M. Heber. Label placement for dynamically generated screen maps. In *Proc. 19th Internat. Cartographic Conf. (ICC'99)*, pages 893–903, Ottawa, Canada, 1999.

[15] S.-H. Poon and C.-S. Shin. Adaptive zooming in point set labeling. In M. Liśkiewicz and R. Reischuk, editors, *Proc. 15th Internat. Sympos. Fundam. Comput. Theory (FCT'05)*, volume 3623 of *Lecture Notes Comput. Sci.*, pages 233–244. Springer-Verlag, 2005.

[16] M. van Kreveld, T. Strijk, and A. Wolff. Point labeling with sliding labels. *Comput. Geom. Theory Appl.*, 13:21–47, 1999.

[17] A. Wolff and T. Strijk. The Map-Labeling Bibliography. `http://i11www.ira.uka.de/map-labeling/bibliography`, 1996.

Appendix

Theorem 2. *There is a polynomial-time reduction from general 1d-ARO with constant dilation for unit-width rectangles (problem (ii)) to general 1d-ARO with proportional dilation (problem (iii)).*

Proof. We transform an instance I of general 1d-ARO with constant dilation for unit-width rectangles into an instance I' of general 1d-ARO with proportional dilation—that is, a set of n unit-width rectangles into a set of n trapezoidal segments of congruent triangles with apexes on the x-axis—such that $H(I) = H(I')$.

For simplicity, we first assume that all vertical edges of rectangles in I lie on distinct vertical lines. We'll remove this restriction later. Let ε be the minimum separation of any two of these vertical lines. The idea is to widen each rectangle into a trapezoid. As long as we widen by no more than $\varepsilon/2$ on either side, we do not introduce any new intersections, and thus do not change the solution. The trick is to ensure that the resulting set of trapezoids forms a valid instance I'.

Let B be the height of the tallest rectangle in I. We first vertically compress I to a height of at most nB by removing empty horizontal strips. We then translate I onto a horizontal strip bounded by the lines $s = s_{\min} := nB/\varepsilon$ and $s = s_{\max} := nB/\varepsilon + nB$. Finally, we embed each rectangle, and its associated trapezoid, in a triangle with apex on the x-axis vertically below the rectangle center and with side edges of slope $\pm 2nB/\varepsilon$, see Figure 13. Each triangle has unit width at $s = s_{\min}$ and width $1 + \varepsilon$ at $s = s_{\max}$. This yields the correctness of our reduction. The reduction is polynomial in the size of a binary encoding of I since B and ε are differences between values specified in the description of I.

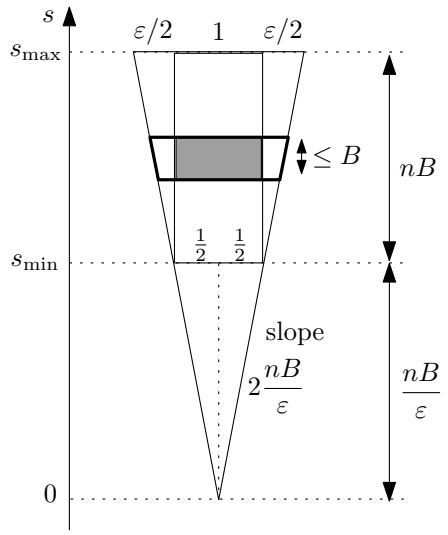


Figure 13: Transforming a rectangle (shaded) into a trapezoidal segment (bold boundary) of a skinny triangle.

If several vertical rectangle edges are collinear, we introduce gaps as follows. Let Δx be the minimum non-zero horizontal distance between any two vertical edges in I, and let $\varepsilon' = \Delta x/n$. Scan I from right to left. For each vertical line ℓ that contains vertical edges of $t > 1$ rectangles $E_1, \ldots, E_t$, numbered lexicographically with respect to their upper left corners, shift each rectangle that lies completely to the right of ℓ by $(t-1)\varepsilon'$, and shift each rectangle E_i to the right by $(i-1)\varepsilon'$. In the end, each pair of vertical edges is separated by at least ε', and, since every rectangle has moved to the right by at most $(n-1)\varepsilon' < \Delta x$, no two have reversed their x-order. □

Theorem 4. *Simple 1d-ARO with proportional dilation can be solved in $O(n^3)$ time using $O(n^2)$ space.*

Proof. Let $\mathcal{E} = \{E_1, \ldots E_n\}$ be the set of extrusions, and let p_i be the apex of wedge-shaped extrusion E_i on the x-axis. For ease of notation define dummy wedges E_0 and E_{n+1} with apexes p_0 and p_{n+1}, and assume

that $p_0, \ldots, p_{n+1}$ are sorted from left to right. For $i < j$, define the *free space* $\Delta(i, j)$ between p_i and p_j to be the triangular or trapezoidal space enclosed by $s = 0$, the right side edge of E_i, the left side edge of E_j, and possibly $s = S_{\max}$. Let $\mathcal{A}[i, j]$ $(i < j)$ be the optimal solution for $p_{i+1}, \ldots, p_{j-1}$ in $\Delta(i, j)$. In $\mathcal{A}[i, j]$, at least one of the truncated extrusions must touch a non-bottom edge of $\Delta(i, j)$, thus dividing the problem into two independent subproblems. For each $k = i + 1, \ldots, j - 1$, we denote by $h_k^{i,j}$ the scale at which E_k first reaches a non-bottom edge of $\Delta(i, j)$. We initialize $\mathcal{A}[i, i + 1] = 0$ for $i = 0, \ldots, n$ and recursively compute

$$\mathcal{A}[i, j] = \max\{\mathcal{A}[i, k] + h_k^{i,j} + \mathcal{A}[k, j] \mid i + 1 \leq k \leq j - 1\}.$$

Obviously, the optimal solution for our problem is $\mathcal{A}[0, n + 1]$. Each of the $O(n^2)$ entries in the dynamic programming table is computed in $O(n)$ time, resulting in a total running time of $O(n^3)$. □

Theorem 5. *The maximum total active range height for a set of n rectangular extrusions of unit height can be approximated within a factor of $1/3$ in $O(n \log n)$ time.*

Proof. For ARO the greedy algorithm for MIS of van Kreveld et al. [16] yields a (1/3)-approximation, for the following reason. Let $\mathcal{S}$ be any optimal solution. If an extrusion E with a non-empty active range in $\mathcal{S}$ is inactive in the solution $\mathcal{A}$ of the algorithm, there must be an extrusion $E' \neq E$ that is selected in $\mathcal{A}$ such that the right edge of E' intersects E; therefore E has been removed from $\mathcal{E}$ in the algorithm and is inactive in $\mathcal{A}$. But since all extrusions are unit-height rectangles and E' has the leftmost right edge when it is selected, E' cannot be responsible for removing from $\mathcal{S}$ rectangles with a total active range height of more than three height units. □

Lemma 1. *The maximum total active range height of a set $\mathcal{E}$ of n rectangles stabbed by a vertical line can be computed in $O(n \log n)$ time.*

Proof. Clearly the active range height of $\mathcal{E}$ equals the height of the union of $\mathcal{E}$. To compute a set of truncated extrusions with this active range height, we sweep the extrusions from top to bottom. We repeatedly pick the topmost extrusion E, set A_E to the current scale, set a_E to s_E, and discard all extrusions E' with $s_{E'} > s_E$. Correctness and running time are obvious. □