Point Location Algorithms of Minimum Size*

Valentina Damerow
vio@uni-paderborn.de

Lukas Finschi
finschi@ifor.math.ethz.ch

Martin Ziegler
ziegler@uni-paderborn.de

Abstract

Consider the classical point location problem: for a fixed arrangement of m hyperplanes and its induced partition of d-space report, upon input of some point, which face it lies in. With sufficient memory, this is easy to solve in logarithmic time $\mathcal{O}(\log m)$. But how fast can algorithms (formalized as Linear Decision Trees) of *minimum size* be? The present work gives lower and upper bounds for the time complexity of point location under this constraint. They show that, in addition to m, the maximum number w of walls of a cell turns out to be a crucial parameter. We also consider a relaxation of the strict minimum-size condition allowing for constant factor overhead.

1 Motivation

Point Location is among the very basic problems in Computational Geometry. Usually, a partition of d-space into regions $F_1, \ldots, F_N$ is fixed and available for preprocessing; the goal is to determine which F_n a given query point $x \in \mathbb{R}^d$ lies in. We consider the case where those regions are the faces induced by hyperplanes; see, e.g., Figure 1.

Definition 1 Fix a non-degenerate[1] arrangement $\mathcal{H} = (H_1, \ldots, H_m)$ of $m > d$ oriented hyperplanes in d-space. For each $H_i^0 := H_i$, let H_i^- and H_i^+ denote its negative and positive halfspace, respectively.

The *position vector* $\pi = \pi_{\mathcal{H}}(x) \in \{-, 0, +\}^m$ indicates, for each i, whether x lies above, on, or below H_i:

$$x \in H_i^{\pi_i} \qquad \forall i = 1, \ldots, m \ .$$

A *face* F_σ of $\mathcal{H}$ is a non-empty collection of points $x \in \mathbb{R}^d$ sharing the same position vector σ:

$$\emptyset \neq F_\sigma = \pi_{\mathcal{H}}^{-1}(\sigma) = \{x \in \mathbb{R}^d : \pi_{\mathcal{H}}(x) = \sigma\}$$

for some $\sigma \in \{-, 0, +\}^m$. Its *dimension* $\dim(F_\sigma)$ equals d minus the number of zeros in σ. □

Figure 1 illustrates that not each possible sign vector σ corresponds to a face: here, no point x satisfies $\pi(x) = (+, +, +)$. Therefore the number of faces does not attain 3^m.

In fact, already Grünbaum [7] observed that in d-space, *any* m non-degenerate hyperplanes yield precisely

$$(1) \qquad N_k = N_k(d, m) = \sum_{i=0}^{k} \binom{d-i}{k-i} \cdot \binom{m}{d-i}$$

k-dimensional faces; a total of $N = \sum_{k=0}^{d} N_k \leq \mathcal{O}(m^d)$.

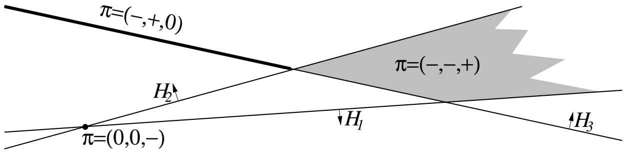

Figure 1: In 2D, three lines induce a total of 19 faces of various dimensions: 3 vertices, 9 walls, 7 cells.

Dobkin and Lipton [6] were the first to achieve logarithmic query time $\mathcal{O}(\log m)$ for this setting. Figure 2 illustrates their approach in 2D: Through each vertex of the arrangement draw a vertical line; locate the 'slab' x lies in using horizontal binary search; within each slab, another vertical binary search reveals the d-dim. face[2] (*cell*) x belongs to.

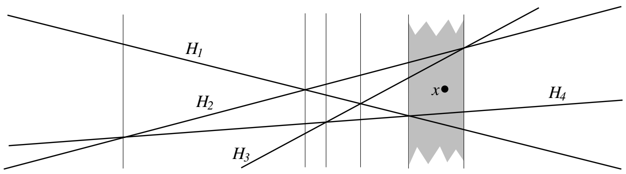

Figure 2: Dobkin and Lipton's additional vertical lines and one induced 'slab'.

In fact, this strategy provides an excess of information: it locates x among the *sub*cells induced by $H_1, \ldots, H_m$ *and* the additional vertical lines $H'_1, \ldots, H'_{m'}$, $m' = \binom{m}{2}$. Further notice that Dobkin and Lipton's algorithm exhibits a rather simple structure: Query, for some hyperplane H, input x's position w.r.t. H; depending on whether x lies left, right of, or on H, choose a next plane and continue. This structure is captured in the following formal model of computation; it is equivalent to Meyer auf der Heide's 3-way LSAs [8] and to Ben-Or's Algebraic Computation Trees [2] restricted to degree-1 polynomials.

Definition 2 (Linear Decision Tree = LDT) Syntactically, an *LDT* T is a rooted ternary tree. Each internal node u is labelled with some oriented hyperplane $H_u \subseteq \mathbb{R}^d$; each leaf v is labelled with some sign vector $\sigma_v \in \{-, 0, +\}^m$.

*Partially supported by DFG Grant Me872/7-3, Heinz Nixdorf Institute, and PaSCo Graduate School

[1] Any d hyperplanes intersect, no $d+1$ of them share a point.

[2] Lower dimensional faces are identified in a similar way, cf. Lemma 12.

The algorithmic semantics: Let r denote T's root. For input $x \in \mathbb{R}^d$, proceed to the left, middle, or right subtree depending on whether x lies on H_r^-, H_r^0, or H_r^+, respectively; continue until reaching a leaf $v = v(x)$.

T *solves* (the point location problem induced by) $\mathcal{H} = (H_1, \ldots, H_m)$ if, for each x, the above process ends up in a leaf labelled with x's position vector: $\sigma_{v(x)} = \pi_{\mathcal{H}}(x)$.

The *size* of an LDT is the number of its leaves. □

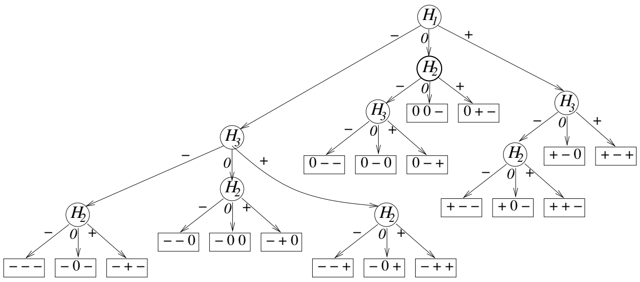

Figure 3: SAMPLE RLDT FOR THE ARRANGEMENT IN Figure 1.

Obviously N, the total number of faces of an arrangement, constitutes a lower bound to the size of any LDT solving the corresponding point location problem: each face has a different position vector and thus requires at least one leaf of its own. An LDT attaining this bound (i.e., having *exactly* N leaves) is said to be of *minimum size*. The one depicted in Figure 3 for instance has this property[3].

Dobkin and Lipton's algorithm however is *not* of minimum size: the introduction of m' additional vertical lines increases the number of cells from $N_2(2, m) = \Theta(m^2)$ to $N' = \Theta(m^3)$, cf. Figure 2. In fact, since their algorithm performs point location within this refined arrangement it has, as an LDT, (at least) N' many leaves.

The natural question arising for LDTs of minimum size is of course: What depth (i.e., running time) can be achieved under this constraint? For answers we refer to Section 2 of the present work.

Some further comments on the above model of computation and its notion of 'size': Any reasonable LDT has, for all its internal nodes u, out-degree either two or three. Therefore the total number of nodes will not differ from its size (in the sense of Definition 2) by more than a factor of two. This in contrast to BRANCHING PROGRAM models (based on directed acyclic graphs rather than trees) where different logical paths of execution are allowed to rejoin; cf. Figure 6.6 in [3].

Actually most algorithms considered 'small' in this model (like [1]) are, as LDTs, *not* of minimum size. The same holds for Fredman's ARITHMETIC MODEL, mostly used to prove lower bounds as in [4]. In this model point location (i.e., halfspace range queries over a trivial semigroup) can be performed in constant time using $\mathcal{O}(m^d)$ storage space which, of course, is not very realistic.

[3] Observe that it queries only hyperplanes belonging to the arrangement. This is no coincidence, cf. Theorem 4.

2 Results

- This work completely characterizes LDTs of minimum size (i.e., algorithms using as few memory as possible).
- It presents upper bounds for their depth (i.e., devises fast algorithms under the above storage constraint)
- and complements them by lower bounds
- both in terms of dimension d, number m of hyperplanes, *and* w, the maximum number of $(d-1)$-dimensional faces (*walls*) any cell of the arrangement has.
- We also investigate on the depth (i.e., running time) that LDTs can achieve under the following relaxed memory constraint: rather than requiring *exactly* size N, grant a constant factor overhead $\mathcal{O}(N)$.

Recall that the LDT shown in Figure 3 uses only hyperplanes from the defining arrangement whereas Dobkin and Lipton's algorithm queries additional vertical lines. LDTs of the first kind are formalized as follows:

Definition 3 (restricted LDT = rLDT) Let $\mathcal{H}$ denote a non-degenerate arrangement of hyperplanes. An LDT solving the associated point location problem is said to be an *rLDT* if it queries only hyperplanes in $\mathcal{H}$. □

Basically, our first result shows that LDTs of minimum size are exactly the restricted ones. This explains the minimality of the LDT from Figure 3 and non-minimality of the one due to Dobkin and Lipton. Roughly speaking, any hyperplane H' queried in addition to those in $\mathcal{H}$ increases the number of possible outcomes, i.e., the number of leaves.

Theorem 4 *Let $\mathcal{H}$ denote a non-degenerate arrangement of hyperplanes. To any rLDT solving $\mathcal{H}$, there exists an rLDT of same depth and minimum size that also solves $\mathcal{H}$. To any minimum-size LDT for $\mathcal{H}$, there exists an rLDT for $\mathcal{H}$ of same depth.* □

Even if Dobkin and Lipton's LDT is larger than necessary, it has a major benefit in terms of speed: a guaranteed depth of $\mathcal{O}(\log m)$ for *any* arrangement of m hyperplanes. Restricted LDTs on the other hand cannot, for certain arrangements, be faster than linear; cf. Figure 4.

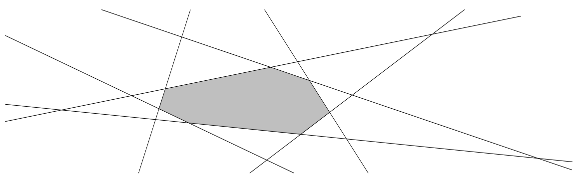

Figure 4: 2D ARRANGEMENT $\mathcal{H}$ WITH $w(\mathcal{H}) = m$: THE SHADED CELL HAS LINEARLY MANY WALLS.

More precisely, the maximum number $w(\mathcal{H})$ of walls of any cell of the arrangement constitutes an important parameter: Any rLDT for $\mathcal{H}$ will need

$$(2) \qquad \text{depth} \quad \geq \quad \Omega\big(w(\mathcal{H}) + d \cdot \log m\big) \ .$$

This has already been observed by Ukkonen [9] and exploited to prove, in this model, exponential lower time bounds for several $\mathcal{NP}$-complete problems. Our second result complements (2) with an upper bound, that is, a fast minimum-size algorithm:

Theorem 5 *To any non-degenerate arrangement $\mathcal{H}$ of m hyperplanes in d-space, there exists an rLDT of*

$$(3) \qquad \text{depth} \quad \leq \quad \mathcal{O}\big(w(\mathcal{H}) \cdot d \cdot \log m\big)$$

solving the associated point location problem. □

Notice that in (3), the '$+$' from (2) is replaced by '$\cdot$' thus leaving a gap between lower and upper bound. Next result nearly closes this gap: In fixed dimension, rLDTs cannot in general be faster than (3) by more than a factor $\log(w)$:

Theorem 6 *To any pair m, w of integers with $m \geq w \geq 3$, there exists a non-degenerate 2D line arrangement $\mathcal{H} = (H_1, \ldots, H_m)$ with $w(\mathcal{H}) = w$ such that any rLDT for $\mathcal{H}$ has*

$$\text{depth} \quad \geq \quad \Omega\big(\tfrac{w}{\log w} \cdot \log m\big) \ . \qquad \square$$

Therefore Theorems 4, 5, and 6 answer almost completely (up to 'small' factors d and $\log w$) the following question:

> *What running times can point location algorithms of exactly minimum size achieve?*

Under this strict constraint slow linear time is, as already pointed out, unavoidable for some arrangements. On the other hand, a small (say, constant factor) memory overhead might very well be acceptable as long as it pays off in terms of improved running time. So what happens if we allow LDTs of size $\mathcal{O}(N)$ rather than exactly N? Notice that Dobkin and Lipton's algorithm still does not match this weakened requirement. And yet, the relaxation indeed pays off: Now, *every* non-degenerate 2D arrangement permits point location in sublinear time as revealed by

Theorem 7 *Let $\mathcal{H}$ be a non-degenerate arrangement of m lines in the plane. There exists an LDT of size $\mathcal{O}(m^2)$ and depth $\mathcal{O}\big(\sqrt[3]{m} \cdot \log m\big)$ solving the associated point location problem.* □

It is not clear whether sublinear time remains feasible in higher dimensions by LDTs of size $\mathcal{O}(m^d)$ nor whether the 2D case can be improved to logarithmic depth.

3 Proofs

Due to space limitations, we present only rough ideas on which our proofs to Theorems 4 to 7 are based. Omitted proofs can be found in the full version.

Definition 8 Let $\mathcal{F}_k(\mathcal{H})$ denote the family of k-dimensional faces of non-degenerate hyperplane arrangement $\mathcal{H}$; let $\mathcal{F}(\mathcal{H}) := \bigcup_{k=0}^{d} \mathcal{F}_k(\mathcal{H})$.

Every $F \in \mathcal{F}(\mathcal{H})$ is a relatively open convex polytope. $\mathcal{H}' \subseteq \mathcal{H}$ will be called a *sub-arrangement*. As $\mathcal{H}$ refines $\mathcal{H}'$, a cell $C \in \mathcal{F}_d(\mathcal{H}')$ may in general consist of several faces from $\mathcal{F}(\mathcal{H})$: it is a *super-cell.* Let $\mathcal{F}_k(\mathcal{H}|_C)$ denote the collection of those faces from $\mathcal{F}_k(\mathcal{H})$ contained in C and $N_k(\mathcal{H}|_C)$ its cardinality. □

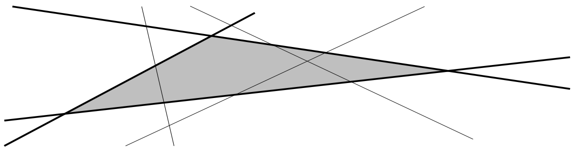

Figure 5: SUPER-CELL CONTAINING 5 ORDINARY CELLS.

3.1 Proof of Theorem 4

For each leaf v of some LDT T for arrangement $\mathcal{H}$, let

$$P(v) \ := \ \big\{x \in \mathbb{R}^d : v(x) = v\big\}$$

denote the set of inputs arriving in v. Those $P(v)$ partition $\mathbb{R}^d$ into relatively open convex polytopes. Furthermore, as T *solves* $\mathcal{H}$, $P(v) \subseteq F_{\sigma(v)}$ by Definition 2: this has already been used in Section 1 to argue that T possesses at least as many leaves as $\mathcal{H}$ has faces. By virtue of the following lemma, the converse inclusion holds as well, provided every leaf of T is reachable:

Lemma 9 *Let T be an rLDT for $\mathcal{H}$ and suppose that $P(v) \neq \emptyset$ for each of its leaves v. Then $P(v) = F_{\sigma(v)}$. In particular, T possesses exactly as many leaves as $\mathcal{H}$ has faces.* □

It can easily be achieved that each leaf of T is reachable: Any 'dead' branch may be pruned without affecting an LDT's operation. According to Lemma 9, a thus stripped rLDT *is* of minimum size.

This time let T be an LDT of minimum size. T itself needs not to be an rLDT: Consider, e.g., a variant T' of the LDT from Figure 3 with the top query for H_2 replaced by any other line H' passing through vertex $\pi^{-1}(0, 0, -)$; as this affects only $x \in H_1^0$, T' is still a valid LDT for $\mathcal{H} = (H_1, H_2, H_3)$ with 19 leaves although $H' \notin \mathcal{H}$.

However it turns out that only this kind of non-restricted queries can occur in minimum size LDTs.

Lemma 10 *Let T be an LDT of minimum size for $\mathcal{H}$. Then any non-restricted query $H' \notin \mathcal{H}$ of T can be replaced by some $H \in \mathcal{H}$ without affecting its operation.* □

Repeated application of this tool obviously converts an LDT of minimum size into an equivalent rLDT of same depth.

3.2 Proof of Theorem 5

The proof consists of two steps. We will first construct an rLDT that only *decides* the arrangement $\mathcal{H}$, i.e., it tells for a given point x whether it lies on any of the hyperplanes or not: "$x \in \bigcup_{H\in\mathcal{H}} H$?" This rLDT can then be extended to one actually *solving* the point location problem, i.e., reporting the face x lies in.

Step number one proceeds by performing *divide-and-conquer* among the putative cells that x might lie in. Recall that for an rLDT we may only query hyperplanes H from $\mathcal{H}$. Intuitively, in order to obtain small depth, H should have about as many cells on its left as on its right. The existence of such a hyperplane is guaranteed by the very central

Lemma 11 *Fix some non-degenerate arrangement $\mathcal{H} = (H_1, \ldots, H_m)$ of hyperplanes in $\mathbb{R}^d$ and a supercell C. Let $w := w(\mathcal{H}|_C)$ denote the maximum number of walls of any cell $c \in \mathcal{F}_d(\mathcal{H}|_C)$. There exists $1 \le i \le m$ such that*

$$\text{(4)} \qquad \frac{N_d(\mathcal{H}|_C) - 1}{w} \;\le\; N_d\big(\mathcal{H}|_{C\cap H_i^-}\big) \;\le\; \frac{N_d(\mathcal{H}|_C)}{2}$$

for an appropriate orientation of H_i. □

Indeed, this result allows for the strategy indicated in Figure 6 which is easily seen to yield an rLDT. Notice that the property "$x \in C$" is maintained throughout the algorithm.

a) Initialize C to be the 'supercell' $\mathbb{R}^d$.
b) Choose $H_i \in \mathcal{H}$ according to Lemma 11;
   query x's position with respect to H_i:
c) if $x \in H_i$, `accept` and exit;
   if $x \in H_i^-$, set $C := C \cap H_i^-$; similarly for $x \in H_i^+$.
d) Continue with b) until C contains only
   a constant number of hyperplanes from $\mathcal{H}$.
e) Check these sequentially:
   if x lies on any of them, `accept`; otherwise `reject`.

Figure 6: RLDT FOR DECIDING $\mathcal{H}$ BASED ON LEMMA 11.

Moreover because of (4), each iteration of the loop body b)-c) reduces the number of cells $c \in \mathcal{F}_d(\mathcal{H}|_C)$ that x might lie in by a factor of nearly $1 - \frac{1}{w}$. Therefore, after at most $\mathcal{O}\big(\log_{1/(1-1/w)} N_d\big)$ iterations, it bails out to e). As $1/\log(\frac{1}{1-1/w}) = \mathcal{O}(w)$ and $N_d := N_d(\mathcal{H}) = \mathcal{O}(m^d)$, this proves a depth of $\mathcal{O}(w \cdot d \cdot \log m)$ for this rLDT *deciding* $\mathcal{H}$.

We will now turn it into an rLDT *solving* the associated point location problem. The underlying construction has already been used by Meyer auf der Heide in a slightly different context to prove CLAIM 3 in [8].

Lemma 12 *Let T be an LDT T of size s deciding the hyperplane arrangement $\mathcal{H}$ in $\mathbb{R}^d$. Then there exists an LDT T' of same depth and* size $\le 2^d \cdot s$ *that solves the associated point location problem. If T is an rLDT, so is T'.* □

The possible factor of 2^d is not significant since restricted LDTs have minimum size anyway, c.f. Theorem 4.

3.3 Proof of Theorem 6

As was shown in the proof to Theorem 4, leaf v of an LDT of minimum size corresponds to a face $P(v)$ of the arrangement $\mathcal{H}$ and vice versa. Let $C \in \mathcal{F}_d(\mathcal{H})$ denote some cell having $w = w(\mathcal{H})$ walls $H_1, \ldots, H_w$ and v its associated leaf of T. $C = P(v)$ being the intersection of all (half)planes queried on the path between v and T's root, $H_1, \ldots, H_w$ must be among these queries. Therefore v is preceded by at least these w query nodes and thus $\operatorname{depth}(T) \ge w$: this is essentially the argument that led Ukkonen to his lower bound (2).

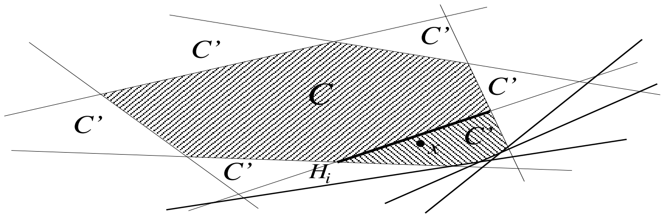

Figure 7: FIRST STEP OF THE ITERATIVE CONSTRUCTION; C ITSELF AND ALL C'S WILL HAVE $w = 6$ WALLS.

A cell C with w walls can be induced by an arrangement $\mathcal{H}_1$ of $m_1 := w$ hyperplanes. Figure 7 indicates how to extend this 2D arrangement by $m_1 \cdot (w-3)$ new lines $\mathcal{H}_2$: Through each of the m_1 triangles adjacent to C place $w-3$ additional lines in such a way that C now is surrounded by cells C' also having w walls each. Notice that each of the added lines also produces a new triangle.

By careful placement it is possible that these lines neither intersect C nor any of the adjacent C'. Thus no cell of $\mathcal{H}_1 \cup \mathcal{H}_2$ has more than w walls: $w(\mathcal{H}_1 \cup \mathcal{H}_2) = w$. Furthermore, $\mathcal{H}_1 \cup \mathcal{H}_2$ consists of $m_2 = m_1 + m_1 \cdot (w-3) \le w^2$ lines in general position. And finally, any LDT of minimum size for $\mathcal{H}_1 \cup \mathcal{H}_2$ has depth at least $2w - 3$:

Consider the leaf v of T with $P(v) = C$. As already pointed out, on the path from T's root to v at least the walls $H_1, \ldots, H_w$ of C must be queried (otherwise it is not possible to distinguish between points lying inside of C or close to C). For a point $x \notin C$ let H_i denote the last tested wall of C, i.e., H_i separates C from one of the C' and assume that x lies in that particular C'. According to our construction C' has w walls, too.

Thus on its way through T, an input $x \in C'$ queries necessarily $H_1, \ldots, H_w$ *and* the $w-3$ additional walls of C', hence the depth is at least $2w-3$. Notice that we did not (and could not) make any presumption on the order in which T would test the walls of C.

Continuing the above construction j times leads to a non-degenerate arrangement $\mathcal{H}_1 \cup \ldots \cup \mathcal{H}_j$ of $m_j = m_{j-1} + (m_{j-1} - m_{j-2}) \cdot (w-3) \leq w^j$ lines where each of the cells $C, C', C'', \ldots, C^{(j)}$ has exactly w walls whereas any other cell does not have more than w walls. Therefore it is $w(\mathcal{H}_1 \cup \ldots \cup \mathcal{H}_j) = w$. Ukkonen's above argument then reveals that an LDT of minimum size for $\mathcal{H}_1 \cup \ldots \cup \mathcal{H}_j$ must have depth at least $w + (j-1) \cdot (w-3) = jw - 3j + 3$. Now choose $j := \log m / \log w$.

3.4 Proof of Theorem 7

Consider a non-degenerate 2D arrangement $\mathcal{H}$ of m lines. If no cell has more than $\mathcal{O}(\sqrt[3]{m})$ walls we are done by virtue of Theorem 5. So suppose that there are 'large' cells. The idea is to introduce further vertical lines that cut these cells into 'smaller' subcells (with less walls). Using this approach Dopkin and Lipton obtain subcells of at most 4 walls each at the expense of $m' = \Theta(m^2)$ additional lines leading to significantly increased size. As mentioned before, for our purpose it suffices to get subcells of at most $\mathcal{O}(\sqrt[3]{m})$ walls each.

Lemma 13 *Let $\mathcal{H}$ denote a non-degenerate arrangement of m lines in the plane. There exist $m' \leq \mathcal{O}(m)$ additional lines $\mathcal{H}'$ such that each cell $C \in \mathcal{F}_d(\mathcal{H} \cup \mathcal{H}')$ has at most $w \leq \mathcal{O}(\sqrt[3]{m})$ walls. Moreover, $\mathcal{H} \cup \mathcal{H}'$ can be chosen as non-degenerate, too.* □

Our proof to Lemma 13 relies heavily on SECTION 3 of [5]. We do not know whether $\mathcal{O}(\sqrt[3]{m})$ is the best (smallest number of walls, maximum taken over all cells) that $m' \leq \mathcal{O}(m)$ additional lines can achieve. In any case it is sufficient to prove Theorem 7: apply Theorem 5 to the arrangement $\mathcal{H} \cup \mathcal{H}'$ with its at most $N' = N(2, m+m') = \mathcal{O}(m^2)$ faces to obtain the claimed LDT for $\mathcal{H}$.

3.5 Proof of Lemma 11

W.l.o.g. orient each hyperplane such that it has no more cells (from $\mathcal{H}|_C$) on its negative than on its positive side, that is,

$$N_d(\mathcal{H}|_{C \cap H_i^-}) \quad \leq \quad N_d(\mathcal{H}|_C)/2 \quad \leq \quad N_d(\mathcal{H}|_{C \cap H_i^+}) \ .$$

We will prove by contradiction that then some H_i^- satisfies (4). So suppose that

$$(5) \quad w \cdot N_d(\mathcal{H}|_{C \cap H_i^-}) < N_d(\mathcal{H}|_C) - 1 \quad \forall i = 1 \ldots m \ .$$

Consider all possible subsets of hyperplanes $\mathcal{H}' \subseteq \mathcal{H}$; for each such $\mathcal{H}'$, consider the collection of those cells c from $\mathcal{H}|_C$ which are, for each $H \in \mathcal{H}'$, on the positive side H^+:

$$(6) \quad \bigcap_{H \in \mathcal{H}'} \mathcal{F}_d(\mathcal{H}|_{C \cap H^+}) \quad = \quad \mathcal{F}_d\Big(\mathcal{H}\Big|_{C \cap \bigcap_{H \in \mathcal{H}'} H^+}\Big) \ .$$

Among those subsets of hyperplanes $\mathcal{H}' \subseteq \mathcal{H}$ for which this collection is non-empty (contains at least one cell) choose some that is maximal w.r.t. set inclusion.

Claim 14 *Such a sub-arrangement $\mathcal{H}'$ has exactly one cell c from $\mathcal{H}|_C$ on the respective positive sides of all $H \in \mathcal{H}'$:*

$$\exists c \in \mathcal{F}_d(\mathcal{H}|_C) \quad : \quad \mathcal{F}_d\Big(\mathcal{H}\Big|_{C \cap \bigcap_{H \in \mathcal{H}'} H^+}\Big) \quad = \quad \{c\} \ .$$

This c thus is a cell not only of $\mathcal{H}$ but also of $\mathcal{H}'$; in other words: each wall H of c belongs to $\mathcal{H}'$, i.e., it has c on its *positive* side. According to the lemma's prerequisite, these are at most w walls: $H_1, \ldots, H_{\tilde{w}}, \tilde{w} \leq w$. And by presumption (5), each of them has strictly less than $(N_d - 1)/w$ cells on its negative side, $N_d := N_d(\mathcal{H}|_C)$. Therefore a total of $< N_d - 1$ cells from $\mathcal{H}|_C$ can have the property of lying on the negative side of any of these walls w from $\mathcal{H}'$.

And by virtue of Claim 14, c is the only cell that fails this property. The total number of cells from $\mathcal{H}|_C$ thus must be $< (N_d - 1) + 1$: contradiction.

Proof (Claim 14): $C \cap \bigcap_{H \in \mathcal{H}'} H^+$ is a super-cell consisting of at least one ordinary cell c from $\mathcal{H}$: we chose $\mathcal{H}'$ such that (6) is non-empty. Now suppose that this (convex!) super-cell consists of several cells from $\mathcal{H}$. Choose an adjacent pair c_1, c_2 and some hyperplane $H \in \mathcal{H} \setminus \mathcal{H}'$ separating them; w.l.o.g. $c_1 \in H^+$ and $c_2 \in H^-$. Then c_1 lies on the positive sides not only of all $H' \in \mathcal{H}'$ but also of this new H which thus can be added to $\mathcal{H}'$ in contradiction to its maximality. □

References

[1] ADAMY, U., R. SEIDEL: "On the Exact Worst Case Query Complexity of Planar Point Location", pp.609-618 in *Proc. 9th ACM-SIAM SODA* (1998).

[2] BEN-OR, M.: "Lower Bounds for Algebraic Computation Trees", pp.80-86 in *Proc. 15th ACM STOC* (1983).

[3] DE BERG, M, M. VAN KREVELD, M. OVERMARS, O. SCHWARZKOPF: "*Computational Geometry: Algorithms and Applications*", Springer (1997).

[4] BRÖNNIMANN, H., B. CHAZELLE, J. PACH: "How Hard is Halfspace Range Searching?", pp.143-155 in *Discr. Comput. Geom.* **10** (1993).

[5] CLARKSON, K.E., H. EDELSBRUNNER, L.J. GUIBAS, M. SHARIR, E. WELZL: "Combinatorial Complexity Bounds for Arrangements of Curves and Spheres", pp.99-160 in *Discr. Comput. Geom.* **5** (1990).

[6] DOBKIN, D., R.J. LIPTON: "Multidimensional Searching Problems", pp.181-186 in *SIAM J. Comput.* **5(2)** (1976).

[7] GRÜNBAUM, B.: "Convex Polytopes", Wiley (1967).

[8] MEYER AUF DER HEIDE, F.: "Fast Algorithms for N-Dimensional Restrictions of Hard Problems", pp.740-747 in *J. ACM* **35(3)** (1988).

[9] UKKONEN, E.: "Exponential Lower Bounds for some $\mathcal{NP}$-complete Problems in Restricted Linear Decision Tree Model", pp.181-192 in *BIT* **23(2)** (1983).