

# On Range Searching in the Group Model and Combinatorial Discrepancy

Kasper Green Larsen  
MADALGO, Department of Computer Science  
Aarhus University  
Denmark  
larsen@cs.au.dk

**Abstract**— In this paper we establish an intimate connection between dynamic range searching in the group model and combinatorial discrepancy. Our result states that, for a broad class of range searching data structures (including all known upper bounds), it must hold that  $t_u t_q = \Omega(\text{disc}^2 / \lg n)$  where  $t_u$  is the worst case update time,  $t_q$  the worst case query time and  $\text{disc}$  is the combinatorial discrepancy of the range searching problem in question. This relation immediately implies a whole range of exceptionally high and near-tight lower bounds for all of the basic range searching problems. We list a few of them in the following:

- For halfspace range searching in  $d$ -dimensional space, we get a lower bound of  $t_u t_q = \Omega(n^{1-1/d} / \lg n)$ . This comes within a  $\lg n \lg \lg n$  factor of the best known upper bound.
- For orthogonal range searching in  $d$ -dimensional space, we get a lower bound of  $t_u t_q = \Omega(\lg^{d-2+\mu(d)} n)$ , where  $\mu(d) > 0$  is some small but strictly positive function of  $d$ .
- For ball range searching in  $d$ -dimensional space, we get a lower bound of  $t_u t_q = \Omega(n^{1-1/d} / \lg n)$ .

We note that the previous highest lower bound for any explicit problem, due to Pătraşcu [STOC'07], states that  $t_q = \Omega((\lg n / \lg(\lg n + t_u))^2)$ , which does however hold for a less restrictive class of data structures.

Our result also has implications for the field of combinatorial discrepancy. Using textbook range searching solutions, we improve on the best known discrepancy upper bound for axis-aligned rectangles in dimensions  $d \geq 3$ .

**Keywords**-range searching; lower bounds; group model; discrepancy; computational geometry;

## 1. INTRODUCTION

Range searching is one of the most fundamental and well-studied topics in the fields of computational geometry and spatial databases. The input to a range searching problem consists of a set of  $n$  geometric objects, most typically points in  $d$ -dimensional space, and the goal is to preprocess the input into a data structure, such that given a query range, one can efficiently aggregate information about the input objects intersecting the query range. Some of the most typical types of query ranges are axis-aligned rectangles, halfspaces, simplices and balls.

The type of information computed over the input objects intersecting a query range include for instance, counting the

number of such objects, reporting them and computing the semi-group or group sum of a set of weights assigned to the objects.

In the somewhat related field of combinatorial discrepancy, the focus lies on understanding set systems. In particular, if  $(Y, \mathcal{A})$  is a set system, where  $Y = \{1, \dots, n\}$  are the elements and  $\mathcal{A} = \{\mathcal{A}_1, \dots, \mathcal{A}_m\}$  is a family of subsets of  $Y$ , then the *minimum discrepancy problem* asks to find a 2-coloring  $\chi : Y \rightarrow \{-1, +1\}$  of the elements in  $Y$ , such that each set in  $\mathcal{A}$  is colored as evenly as possible, i.e. find  $\chi$  minimizing  $\text{disc}(\chi, Y, \mathcal{A})$ , where

$$\text{disc}(\chi, Y, \mathcal{A}) = \max_j \left| \sum_{i \in \mathcal{A}_j} \chi(i) \right|.$$

The main result of this paper is the establishment of an intimate connection between dynamic range searching in the group model and combinatorial discrepancy. Our results have strong implications for both fields. For range searching, we obtain exceptionally high and near-tight lower bounds for all of the basic problems, and for combinatorial discrepancy we improve the best upper bounds for one of the most well-studied problems using textbook range searching solutions.

### 1.1. Range Searching in the Group Model

In this paper, we focus on dynamic range searching in the group model. In this setting, each input object to a range searching problem is assigned a weight from a commutative group, and the goal is to preprocess an input set into a data structure, consisting of a collection of group elements and auxiliary data, such that given a query range, one can efficiently compute the group sum of the weights assigned to the input objects intersecting the query range. The data structure answers queries by adding and subtracting a subset of the precomputed group elements (in a group, each element has an inverse element, thus we have subtraction) to finally yield the answer to the query. In addition to answering queries, we require that a data structure supports updating the weights of the input objects.

Since the group model was first introduced there has been two slightly different definitions of data structures in this model, one a bit less restrictive than the other. The most restrictive type of data structure is known in the literature

Kasper Green Larsen is supported in part by a Google Europe Fellowship in Search and Information Retrieval, and in part by MADALGO – Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation.

as *oblivious*, while the other type has not received a formal name. To make the difference clear, we have chosen to name this other type of data structure *weakly oblivious*. In the following we review both definitions, starting with the least restrictive:

*Weakly Oblivious Data Structures:* A *weakly oblivious* data structure in the group model, is a dynamic data structure with no understanding of the particular group in question, i.e. it can only access and manipulate weights through black-box addition and subtraction [15]. Thus from the data structure’s point of view, each precomputed group element is just a linear combination over the weights (and possibly previous weights) assigned to the input objects. When answering a query, such a data structure adds and subtracts a subset of these linear combinations to finally yield the linear combination summing exactly the weights currently assigned to the input objects intersecting the query range. When given an update request, the data structure may delete some of the stored group elements, and also create new group elements to store by adding and subtracting both previously stored values and the newly assigned weight.

The query time of such a data structure is defined as the number of precomputed group elements used when answering a query, and the update time is defined as the number of created and deleted group elements on an update request. Thus deciding which precomputed group elements to add and subtract is considered free of charge.

We note that if we did not require the data structure to have no knowledge of the group in question, then range searching over any finite group would be trivial: The data structure could simply encode the weights assigned to the input points in the auxiliary data, and thus compute the answer to a query using no group operations at all. The group model is thus incomparable to the classic cell-probe model.

*Oblivious Data Structures:* The second, and slightly more restrictive definition of data structures, was given by Fredman [16]. Again data structures are considered to have no knowledge of the group, and queries are still answered by adding and subtracting precomputed linear combinations over the weights assigned to the input points. The update operations are however more constrained: an update of the weight assigned to an input object  $p$ , is supported simply by re-evaluating every precomputed group element for which the weight of  $p$  occurs with non-zero coefficient in the corresponding linear combination. Every stored group element thus corresponds to a linear combination over the currently assigned weights, and may not include previous weights. We refer to such data structures as *oblivious* data structures.

We define the query time of an oblivious data structure as the number of group elements used when answering a query, and the update time is defined as the number of linear combinations that need to be re-evaluated when

updating the weight of an input object. We note that lower bounds proved for weakly oblivious data structures also apply to oblivious data structures. For a more formal and mathematical definition of an oblivious data structure, we refer the reader to Section 2.

Given that data structures in the group model have no understanding of the particular group in question, the additional freedom allowed in the weakly oblivious group model might seem artificial. Thus we note that the main motivating factor for studying the weakly oblivious model, is that previous lower bounds proved in this model were (somewhat) easily extendible to cell probe lower bounds.

*Previous Results:* In the following, we first review the previous results on lower bounds for range searching problems in the (semi-group and) group model, and then present the best known upper bounds for the two most fundamental range searching problems: orthogonal range searching and halfspace range searching.

In the related semi-group model, researchers have been very successful in proving lower bounds. In the semi-group model, input objects have been assigned a weight from a commutative semi-group (elements do not necessarily have inverse elements), and the goal is to compute the semi-group sum of the weights assigned to input objects intersecting a query range. Since there are no inverse elements, a data structure cannot subtract. This gives a very geometric flavor to range searching lower bound proofs: If a data structure stores a precomputed semi-group element involving the weight of an input object, then the query algorithm can only use that precomputed group element when answering query ranges that intersects that input object (its weight cannot be cancelled out). Thus semi-group lower bound proofs boils down to arguing that it is hard to “cover” all query ranges with a small collection of subsets of input objects.

Unfortunately we have no such property when allowing subtraction (i.e. the group model). The difficulties encountered when moving from the semi-group to the group model have been recognized as major obstacles for decades, and we believe the following quote by Pătraşcu captures the essence of these difficulties:

“Philosophically speaking, the difference in the type of reasoning behind semi-group lower bounds and group lower bounds is parallel to the difference between *understanding geometry* and *understanding computation*. Since we have been vastly more successful at the former, it should not come as a surprise that progress outside the semi-group model has been extremely slow [20].”

In 1982, Fredman [16] gave the definition of an oblivious data structure in the group model. Under this definition, he managed to prove an  $\Omega(\lg n)$  lower bound on the average cost per operation in a sequence of  $n$  updates and  $n$  queries to the *partial sums* problem. In the partial sums problem, the input is an array of  $n$  entries, each storing an element from a commutative group, and the goal is to support weight

updates and range queries of the form: “What is the group sum of the elements in the subarray from index  $i$  through  $j$ ?”.

The next result on group model lower bounds was due to Fredman and Saks [15], who introduced the celebrated chronogram technique. Using this technique, they again proved lower bounds for the partial sums problem, stating that any dynamic data structure must have an average cost per operation of  $\Omega(\lg n / \lg \lg n)$  over a sequence of  $n$  queries and  $n$  updates [15]. While the lower bound is weaker than the earlier lower bound of Fredman, it holds also for weakly oblivious data structures.

In [12] and [11] Chazelle proved lower bounds for *offline* range searching in the group model. For the problem of offline halfspace range searching in two-dimensional space, he showed an  $\Omega(n \lg n)$  lower bound on the total number of group operations needed [12]. In the offline halfspace range searching problem, the input consists of  $n$  points and  $n$  halfspaces, and the goal is to compute for each halfspace the group sum of the weights assigned to the points intersecting it. In [11] he applied his techniques to orthogonal range searching as well, and managed to show a lower bound of  $\Omega(n \lg \lg n)$  in the two-dimensional case. Orthogonal range searching is essentially the extension of partial sums to higher dimensions: Here the input consists of  $n$  points, and the query ranges are axis-aligned rectangles.

The next big result was due to Pătraşcu and Demaine [21], who managed to show an  $\Omega(\lg n)$  lower bound on the average cost per operation over a sequence of  $n$  updates and  $n$  queries to the partial sums problem. While matching the early results of Fredman, this bound also holds for weakly oblivious data structures.

Finally, Pătraşcu [20] proved an  $\Omega(\lg n / \lg(\lg n + S/n))$  lower bound for the query time of *static* data structures for two-dimensional orthogonal range searching. Here  $S$  is the space used by the data structure in number of precomputed group sums. Using an elegant extension of the chronogram technique, this provided the highest lower bound to date for any dynamic range searching problem in the group model, namely  $t_q = \Omega((\lg n / \lg(\lg n + t_u))^2)$ , where  $t_q$  is the query time and  $t_u$  is the update time. This lower bound applies to weakly oblivious data structures for two-dimensional orthogonal range searching.

On the upper bound side, there is no separation between what has been achieved for oblivious and weakly oblivious data structures. Thus, all the bounds we mention in the following hold for both types of data structures.

The best results for  $d$ -dimensional orthogonal range searching in the group model is achieved through the data structures known as *range trees* [8]. These data structures provide a solution with  $t_q = t_u = O(\lg^d n)$ . From the above lower bounds, these data structures are seen to be optimal in 1-d, and to have a query time within  $\lg^{O(1)} \lg n$  factors from optimal in 2-d. Unfortunately it is not even known from a

lower bound perspective whether the query and update time must grow with dimension.

For halfspace range searching, one can use Chan’s latest results on *partition trees* to give data structures with  $t_u = O(\lg \lg n)$  and  $t_q = O(n^{1-1/d})$  [10], and with some extra work, one can extend the results in [19] to achieve a tradeoff between query time and update time of  $t_q = \tilde{O}(n^{1-1/d}/t_u^{1/d})$ , for any  $t_u = \Omega(\lg n)$ . Here  $\tilde{O}(\cdot)$  hides polylogarithmic factors. Thus the highest known lower bound for any explicit problem is exponentially far from the best known upper for halfspace range searching.

## 1.2. Combinatorial Discrepancy

The minimum discrepancy problem mentioned earlier is the central problem in combinatorial discrepancy. Understanding the best achievable colorings for various families of set systems has been an active line of research for decades, and the results obtained have found numerous applications in other areas of computer science, see for instance the seminal books of Matoušek [18] and Chazelle [13] for introductions to discrepancy theory, and for applications in complexity lower bounds, computational geometry, pseudo-randomness and communication complexity.

The results most important to our work are those related to families of set systems with a range searching flavor to them. More formally, if we let  $X$  be a universe of geometric objects (think of  $X$  as all possible input geometric objects to a range searching problem, for instance all points  $d$ -dimensional space),  $P \subset X$  a set of  $n$  geometric input objects (a concrete input to a range searching problem) and  $\mathcal{R}$  a collection of query ranges, where each query range  $R \in \mathcal{R}$  is a subset of  $X$  (for every query to the range searching problem,  $\mathcal{R}$  contains a set consisting of those elements in  $X$  that intersects the query range), then we define the *induced set system*  $(P, \mathcal{A}_{P, \mathcal{R}})$ , where  $\mathcal{A}_{P, \mathcal{R}} = \{R \cap P : R \in \mathcal{R}\}$  is the family of sets containing for each  $R \in \mathcal{R}$ , the set consisting of all input objects that are contained in  $R$  ( $\mathcal{A}_{P, \mathcal{R}}$  is also known in the literature as the trace of  $\mathcal{R}$  on  $P$ ). With this definition, we define the *discrepancy*  $\text{disc}(P, \mathcal{R})$  as

$$\text{disc}(P, \mathcal{R}) = \min_{\chi: P \rightarrow \{-1, +1\}} \text{disc}(\chi, P, \mathcal{A}_{P, \mathcal{R}}),$$

thus the discrepancy measures the best achievable 2-coloring of the induced set system of  $P$  and  $\mathcal{R}$ . Finally, we define the number of *distinct query ranges* of an induced set system as  $|\mathcal{A}_{P, \mathcal{R}}|$ , that is, as the number of distinct sets ( $\mathcal{A}_{P, \mathcal{R}}$  is not a multiset).

To make absolutely clear the connection to range searching, consider as an example the  $d$ -dimensional orthogonal range searching problem. Here  $X$  is simply  $\mathbb{R}^d$ , i.e. the set of all  $d$ -dimensional points. The family  $\mathcal{R}$  similarly contains all axis-aligned rectangles in  $\mathbb{R}^d$  (each axis-aligned rectangle is a subset of  $\mathbb{R}^d$ ). Finally, we see that for any induced set system  $(P, \mathcal{A}_{P, \mathcal{R}})$ , where  $P$  is a set of  $n$  input points in  $X$ ,

the number of distinct query ranges is bounded by  $O(n^{2d})$  since each axis-aligned rectangle defining a range in  $\mathcal{R}$  can be shrunk to have one point from  $P$  on each of its  $2d$  sides without changing the set of input points contained in the query range.

*Previous Results:* In the following, we review the discrepancy upper and lower bounds related to the most fundamental types of range searching. We note that a lower bound on the discrepancy of a range searching problem is a proof that *there exists* a subset  $P \subset X$  of  $n$  input objects to the range searching problem, for which  $\text{disc}(P, \mathcal{R})$  is bounded from below, while upper bounds on the discrepancy imply that  $\text{disc}(P, \mathcal{R})$  is bounded from above *for all* subsets  $P \subset X$  of  $n$  input objects.

The discrepancy of halfspace range searching is particularly well understood. If we let  $\mathcal{H}_d$  denote the set of all halfspaces in  $d$ -dimensional space (where each halfspace  $H \in \mathcal{H}_d$  is a subset of  $\mathbb{R}^d$ ), then Alexander [1] proved that there exists a set  $P$  of  $n$  points in  $\mathbb{R}^d$ , such that  $\text{disc}(P, \mathcal{H}_d) = \Omega(n^{1/2-1/2d})$ . A matching upper bound was subsequently established by Matoušek [17].

For orthogonal range searching (or axis-aligned rectangles), the picture is more muddy. On the lower bound side, Beck [4] proved that there exists a set  $P$  of  $n$  points in  $\mathbb{R}^2$ , such that  $\text{disc}(P, \mathcal{B}_2) = \Omega(\lg n)$ , where we use  $\mathcal{B}_d$  to denote the family containing all axis-aligned rectangles in  $\mathbb{R}^d$ . However, in dimensions  $d \geq 3$ , the highest achieved lower bound is only  $\Omega(\lg^{(d-1)/2+\mu(d)} n)$ , where  $\mu(d) > 0$  is some small but strictly positive function of  $d$  [9]. On the upper bound side, Srinivasan [23] proved that  $\text{disc}(P, \mathcal{B}_2) = O(\lg^{2.5} n)$  for any set  $P$  of  $n$  points in  $\mathbb{R}^2$ , and for dimensions  $d \geq 3$ , the best upper bound is  $O(\lg^{d+1/2} n \sqrt{\lg \lg n})$  [18].

If the ranges are balls with arbitrary radius, then a discrepancy lower bound of  $\Omega(n^{1/2-1/2d})$  can be established from the results on halfspace ranges [18] (a large enough ball looks locally like a halfspace). A matching lower bound was proved in the two-dimensional case, even when all balls (discs) have a fixed radius [5].

For line range searching, Chazelle and Lvov proved that there exists a set  $P$  of  $n$  points in  $\mathbb{R}^2$ , such that  $\text{disc}(P, \mathcal{L}_2) = \Omega(n^{1/6})$  [14]. Here  $\mathcal{L}_2$  denotes the set of all lines in two-dimensional space.

Another interesting lower bound is related to arithmetic progressions. Let  $(Y, \mathcal{A})$  be the set system where  $Y = \{0, \dots, n-1\}$  and  $\mathcal{A}$  contains every arithmetic progression on  $Y$ , i.e. for every pair of integers  $i, d$  satisfying  $0 \leq i, d < n$ ,  $\mathcal{A}$  contains the set  $\mathcal{A}_{i,d} = \{i+jd \mid j \in \{0, \dots, \lfloor (n-i-1)/d \rfloor\}\}$ . Then Roth proved  $\text{disc}(Y, \mathcal{A}) = \Omega(n^{1/4})$  [22].

Finally, we conclude by mentioning some additional discrepancy upper bounds that are related to the later proofs in this paper. If  $(Y, \mathcal{A})$  is a set system in which every  $\mathcal{A}_i \in \mathcal{A}$  has cardinality at most  $t$ , then Banaszczyk [2] proved that  $\text{disc}(Y, \mathcal{A}) = O(\sqrt{t \lg |Y|})$ . The best bound for this problem that is independent of  $|Y|$  is due to Beck and Fiala [6], and

it states that  $\text{disc}(Y, \mathcal{A}) = O(t)$ . We note that there exist results [7] improving on the additive constants in the bound of Beck and Fiala. While many discrepancy upper bounds are purely existential, we mention that Bansal [3] recently gave constructive discrepancy minimization algorithms for several central problems.

### 1.3. Our Results

The main result of this paper is the establishment of a theorem relating the update and query time of dynamic range searching data structures in the group model and the combinatorial discrepancy of the corresponding range searching problem. Before presenting our theorem, we need one final definition regarding oblivious data structures.

Recall that an oblivious data structure preprocesses an input set of  $n$  geometric objects into a collection of group elements, each corresponding to a linear combination over the weights assigned to the input objects. Queries are answered by again computing linear combinations over the precomputed group elements, and updates are supported by re-evaluating every linear combination involving the weight of the updated point. We define the *multiplicity* of an oblivious data structure as the largest absolute value occurring as a coefficient in any of these linear combinations. We note that every known data structure uses only coefficients amongst  $\{-1, 0, +1\}$ , thus all known data structures have multiplicity 1, but there is nothing inherent in the group model that prevents larger coefficients. When giving our more formal definition of oblivious data structures in Section 2, we also give a more precise definition of multiplicity.

We are finally ready to present the main result of this paper:

**Theorem 1.** *Let  $\mathcal{R}$  be the query ranges of a range searching problem, where each set in  $\mathcal{R}$  is a subset of a universe  $X$ . Furthermore, let  $P \subset X$  be a set of  $n$  geometric input objects to the range searching problem. Then any oblivious data structure for the range searching problem must satisfy  $\Delta^2 \sqrt{t_u t_q \lg m} = \Omega(\text{disc}(P, \mathcal{R}))$  on the input set  $P$ . Here  $\Delta$  denotes the multiplicity of the data structure,  $t_u$  its worst case update time,  $t_q$  its worst case query time and  $m$  the number of distinct query ranges in  $(P, \mathcal{A}_{P, \mathcal{R}})$ .*

Thus for constant multiplicity oblivious data structures (which includes all known upper bounds), we get extremely high lower bounds compared to previous results. We mention these lower bounds in the following (for constant multiplicity), and note that the number of distinct query ranges for all of the considered problems is polynomial in the input size:

For halfspace range searching in  $d$ -dimensional space we get a lower bound of

$$t_u t_q = \Omega(n^{1-1/d} / \lg n),$$

simply by plugging in the discrepancy lower bound of  $\Omega(n^{1/2-1/2d})$ . This comes within a  $\lg n \lg \lg n$  factor of Chan's upper bound, and is exponentially larger than the highest previous lower bound for any explicit problem of  $t_q = \Omega((\lg n / \lg(\lg n + t_u))^2)$ . We note that halfspace range searching is a special case of simplex range searching, this bound therefore also applies to simplex range searching.

For orthogonal range searching, we do not improve on the best bounds in the two-dimensional case, but for  $d$ -dimensional orthogonal range searching we get a lower bound of

$$t_u t_q = \Omega\left(\lg^{d-2+2\mu(d)} n\right)$$

from the discrepancy lower bound  $\Omega(\lg^{(d-1)/2+\mu(d)} n)$ . By a simple reduction, this bound also applies to the well-studied problem of  $d$ -dimensional rectangle stabbing (range searching where the input set contains axis-aligned rectangles, and the queries are points).

For  $d$ -dimensional ball range searching, our lower bound matches that for halfspace range searching, and in the two-dimensional case the bound holds even when all query balls (discs) have the same fixed radius.

For line range searching, that is, range searching where the input is a set of  $n$  two-dimensional points and a query ask to sum the weights of all points intersecting a query line, we get a lower bound of  $t_u t_q = \Omega(n^{1/3} / \lg n)$ .

Finally, for the arithmetic progression range searching problem, i.e. the range searching problem where the input is a set of  $n$  ordered points  $p_0, \dots, p_{n-1}$  and a query asks to sum the weights of the points in an arithmetic progression (see Section 1.2), we get a lower bound of  $t_u t_q = \Omega(n^{1/2} / \lg n)$ .

For more lower bounds we refer the reader to the books by Matoušek [18] and Chazelle [13].

Our result also has implications for the field of combinatorial discrepancy. By contraposition of Theorem 1, we get a discrepancy upper bound for  $d$ -dimensional orthogonal range searching (axis-aligned rectangles) of  $O(\lg^{d+1/2} n)$  directly from the textbook range tree data structures with  $t_u = t_q = O(\lg^d n)$ . While the improvement over the best previous result is only a  $\sqrt{\lg \lg n}$  factor in dimensions  $d \geq 3$ , we still find this a beautiful example of the interplay between data structures and combinatorial discrepancy.

Finally, we mention that our proof of Theorem 1 relies on a, we believe, novel application of discrepancy upper bound techniques.

## 2. PRELIMINARIES

In the following we define range searching, oblivious data structures and discrepancy in terms of matrices.

*Incidence Matrices:* Let  $(P, \mathcal{A}_{P, \mathcal{R}})$  be the induced set system of a set  $P = \{p_1, \dots, p_n\}$  of  $n$  geometric objects and a family  $\mathcal{R}$  of query ranges. Then we define the *incidence matrix*  $\mathcal{C}_{P, \mathcal{R}} \in \{0, 1\}^{|\mathcal{A}_{P, \mathcal{R}}| \times n}$  of  $\mathcal{R}$  and  $P$

as the  $\{0, 1\}$ -matrix having a column for each input object in  $P$  and a row for each set in the induced set system  $\mathcal{A}_{P, \mathcal{R}} = \{\mathcal{A}_1, \dots, \mathcal{A}_{|\mathcal{A}_{P, \mathcal{R}}|}\}$ . The  $i$ 'th row of  $\mathcal{C}_{P, \mathcal{R}}$  has a 1 in the  $j$ 'th column if  $p_j \in \mathcal{A}_i$  and a 0 otherwise.

*Oblivious Data Structures:* Consider a range searching problem where the query ranges  $\mathcal{R}$  are subsets of a universe  $X$ . Then an oblivious data structure for the range searching problem is a factorization of each incidence matrix  $\mathcal{C}_{P, \mathcal{R}}$ , where  $P \subset X$  is a set of  $n$  geometric input objects, into two matrices  $Q_{P, \mathcal{R}}$  and  $D_{P, \mathcal{R}}$  such that  $Q_{P, \mathcal{R}} \cdot D_{P, \mathcal{R}} = \mathcal{C}_{P, \mathcal{R}}$  [16].

The *data matrix*  $D_{P, \mathcal{R}} \in \mathbb{Z}^{S \times n}$  represents the precomputed group sums stored by the data structure on input  $P$ . Each of the  $S$  rows is interpreted as a linear combination over the weights assigned to the  $n$  input objects, and we think of the data structure as maintaining the corresponding group sums when given an assignment of weights to the input objects.

The *query matrix*  $Q_{P, \mathcal{R}} \in \mathbb{Z}^{|\mathcal{A}_{P, \mathcal{R}}| \times S}$  specifies the query algorithm. It has one row for each set  $\mathcal{A}_i$  in the induced set system  $\mathcal{A}_{P, \mathcal{R}}$ , and we interpret this row as a linear combination over the precomputed group sums, denoting which elements to add and subtract when answering a query range intersecting precisely the input objects in  $\mathcal{A}_i$ .

We note that with the above interpretations of the data and query matrix, we get that  $Q_{P, \mathcal{R}} \cdot D_{P, \mathcal{R}} = \mathcal{C}_{P, \mathcal{R}}$  ensures that when given a query range  $R \in \mathcal{R}$ , the query algorithm adds and subtracts a subset of the precomputed linear combinations to finally yield the linear combination summing precisely the weights assigned to the input objects intersecting  $R$ . For a concrete example of what the matrices corresponding to a data structure might look like, we refer the reader to Section 4.1 where we review the data structure solution for orthogonal range searching.

The *worst case query time* of an oblivious data structure on an input set  $P$ , is defined as the maximum number of non-zero entries in a row of  $Q_{P, \mathcal{R}}$ . The *worst case update time* on an input set  $P$  is similarly defined as the maximum number of non-zero entries in a column of  $D_{P, \mathcal{R}}$ . The *space* of the data structure is the number of columns in  $Q_{P, \mathcal{R}}$  (equivalently number of rows in  $D_{P, \mathcal{R}}$ ). Finally, we define the *multiplicity* as the largest absolute value of an entry in  $D_{P, \mathcal{R}}$  and  $Q_{P, \mathcal{R}}$ .

*Combinatorial Discrepancy and Matrices:* The definition of discrepancy can also be stated in terms of matrices. Let  $P$  be a set of  $n$  geometric input objects and  $\mathcal{R}$  a family of query ranges. Then

$$\text{disc}(P, \mathcal{R}) = \min_{x \in \{-1, +1\}^n} \|\mathcal{C}_{P, \mathcal{R}} \cdot x\|_\infty$$

is easily seen to be the exact same definition of  $\text{disc}(P, \mathcal{R})$  as the one we presented in the introduction. Here  $\|\cdot\|_\infty$  gives the largest absolute value amongst the coordinates of a vector.

### 3. ESTABLISHING THE CONNECTION

In this section we prove our main result, Theorem 1. Let  $\mathcal{R}$  be a collection of query ranges, all subsets of a universe  $X$ . Also let  $P \subset X$  be a set of  $n$  geometric input objects. Our goal is to show that  $\mathcal{C}_{P,\mathcal{R}}$  cannot be factored into two matrices  $Q_{P,\mathcal{R}}$  and  $D_{P,\mathcal{R}}$ , unless  $Q_{P,\mathcal{R}}$  has a row with many non-zero entries, or  $D_{P,\mathcal{R}}$  has a column with many non-zero entries, i.e. either the query or update time of an oblivious data structure for the input set  $P$  must be high.

Our key idea for proving this, is to multiply a factorization by a cleverly chosen vector. More formally, if  $Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} = \mathcal{C}_{P,\mathcal{R}}$  is a factorization provided by the oblivious data structure, then we find a vector  $x \in \mathbb{R}^n$  such that  $Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} \cdot x$  has small coefficients if  $Q_{P,\mathcal{R}}$  and  $D_{P,\mathcal{R}}$  are too sparse, and at the same time  $\mathcal{C}_{P,\mathcal{R}} \cdot x$  has large coefficients. Since  $Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} \cdot x = \mathcal{C}_{P,\mathcal{R}} \cdot x$  this gives us our lower bound. The trick in finding a suitable vector  $x$  is to consider vectors in  $\{-1, +1\}^n$ . Making this restriction immediately allows us to use combinatorial discrepancy lower bounds to argue that  $\mathcal{C}_{P,\mathcal{R}} \cdot x$  has large coefficients, and at the same time we can use combinatorial discrepancy upper bound techniques to exploit the sparse rows and columns of  $Q_{P,\mathcal{R}}$  and  $D_{P,\mathcal{R}}$ .

*Proof of Theorem 1:* Let  $\mathcal{R}$  be the query ranges of the range searching problem and  $P$  a set of  $n$  geometric input objects. Furthermore, let  $m$  denote the number of distinct query ranges in  $(P, \mathcal{A}_{P,\mathcal{R}})$ , and assume that an oblivious data structure for the input set  $P$  exists, having worst case update time  $t_u$ , worst case query time  $t_q$  and multiplicity  $\Delta$ .

Let  $Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} = \mathcal{C}_{P,\mathcal{R}}$  denote the corresponding factorization provided by the oblivious data structure. Our first step is to argue that there exists a vector  $x \in \{-1, +1\}^n$  for which  $\|Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} \cdot x\|_\infty$  is small. The existence of this vector is guaranteed by the following theorem:

**Theorem 2.** *Let  $Q \in \mathbb{R}^{m \times p}$  and  $D \in \mathbb{R}^{p \times n}$  be two matrices of reals, such that every row of  $Q$  has at most  $t_Q$  non-zero entries, and every column of  $D$  has at most  $t_D$  non-zero entries. Finally, let  $\Delta$  be an upper bound on the absolute value of any entry in  $Q$  and  $D$ . Then there exists a vector  $x \in \{-1, +1\}^n$ , such that  $\|QDx\|_\infty = O(\Delta^2 \sqrt{t_D t_Q \lg m})$ .*

Before proving the theorem, we show that it implies Theorem 1. Recall that all coefficients in  $Q_{P,\mathcal{R}}$  and  $D_{P,\mathcal{R}}$  are bounded in absolute value by the multiplicity  $\Delta$  of the oblivious data structure. At the same time, each row of  $Q_{P,\mathcal{R}}$  has at most  $t_q$  non-zero entries, and each column of  $D_{P,\mathcal{R}}$  has at most  $t_u$  non-zero entries. Finally, since  $(P, \mathcal{A}_{P,\mathcal{R}})$  has at most  $m$  distinct query ranges, we get that  $Q_{P,\mathcal{R}}$  has at most  $m$  rows. Thus by Theorem 2, there exists a vector  $x^* \in \{-1, +1\}^n$  such that  $\|Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} \cdot x^*\|_\infty = O(\Delta^2 \sqrt{t_u t_q \lg m})$ . Since  $x^* \in \{-1, +1\}^n$ , we also have  $\|\mathcal{C}_{P,\mathcal{R}} \cdot x^*\|_\infty \geq \text{disc}(P, \mathcal{R})$ . But  $Q_{P,\mathcal{R}} \cdot D_{P,\mathcal{R}} \cdot x^* = \mathcal{C}_{P,\mathcal{R}} \cdot x^*$ , thus it must hold that  $\Delta^2 \sqrt{t_u t_q \lg m} = \Omega(\text{disc}(P, \mathcal{R}))$ , which completes the proof of Theorem 1.

*Proof of Theorem 2:* This paragraph is devoted to proving Theorem 2. Throughout the paragraph, we let  $Q \in \mathbb{R}^{m \times p}$  and  $D \in \mathbb{R}^{p \times n}$  be matrices satisfying the constraints of Theorem 2. The main tool in our proof is a result in discrepancy theory due to Banaszczyk [2]. We first introduce some terminology and then present his result.

Let  $\|\cdot\|_2$  denote the Euclidian norm on  $\mathbb{R}^p$ , and  $B_2^p$  the closed Euclidean unit ball in  $\mathbb{R}^p$ . Let  $\gamma_p$  denote the (standard)  $p$ -dimensional Gaussian measure on  $\mathbb{R}^p$  with density  $(2\pi)^{-p/2} e^{-\|x\|_2^2/2}$ . Then the following holds

**Theorem 3** (Banaszczyk [2]). *There is a numerical constant  $c > 0$  with the following property. Let  $K$  be a convex body in  $\mathbb{R}^p$  with  $\gamma_p(K) \geq 1/2$ . Then to each sequence  $u_1, \dots, u_n \in cB_2^p$  there corresponds signs  $\varepsilon_1, \dots, \varepsilon_n \in \{-1, +1\}$  such that  $\varepsilon_1 u_1 + \dots + \varepsilon_n u_n \in K$ .*

To prove Theorem 2, we seek a column vector  $x \in \{-1, +1\}^n$  that somehow simultaneously exploits the sparse rows of  $Q$  and the sparse columns of  $D$ . We argue for the existence of this vector by carefully defining a convex body capturing the sparsity of  $Q$ , and a sequence of vectors in  $cB_2^p$  that captures the sparsity of  $D$ . The application of Theorem 3 on this choice of convex body and vectors in  $cB_2^p$  then yields the desired vector  $x$ . We first define the following convex body  $K_\alpha$  in  $\mathbb{R}^p$ :

$$K_\alpha := \{y = (y_1, \dots, y_p) \in \mathbb{R}^p \mid |\langle Q_1, y \rangle| \leq \alpha \wedge \dots \wedge |\langle Q_m, y \rangle| \leq \alpha\}$$

where  $Q_i$  denotes the  $i$ 'th row vector of  $Q$ ,  $\langle Q_i, y \rangle = \sum_{j=1}^p Q_{i,j} y_j$  is the standard inner product, and  $\alpha \geq 0$  is a parameter to be fixed later.

In understanding our choice of  $K_\alpha$ , think of each dimension in  $\mathbb{R}^p$  as representing one coordinate of  $Dx$ . In this setting, each of the constraints  $|\langle Q_i, y \rangle| \leq \alpha$  intuitively forces the coordinates of  $QDx$  to be small. Our goal is to employ Theorem 3 on  $K_\alpha$ , thus we find a value of  $\alpha$  such that  $\gamma_p(K_\alpha) \geq 1/2$ :

**Lemma 1.** *If  $\alpha = \Omega(\Delta \sqrt{t_Q \lg m})$ , then  $\gamma_p(K_\alpha) \geq 1/2$ .*

*Proof:* Recall that  $\gamma_p(K_\alpha)$  denotes the probability that a random vector  $z \in \mathbb{R}^p$ , with each coordinate distributed independently as a Gaussian with mean 0 and variance 1, lies within  $K_\alpha$ . In computing this probability, we first bound  $\Pr[|\langle Q_i, z \rangle| > \alpha]$  for a fixed  $i$ . Since each row  $Q_i$  has at most  $t_Q$  non-zero entries, we get that  $\langle Q_i, z \rangle$  is a linear combination of at most  $t_Q$  independent Gaussians, each with mean 0 and variance 1. Furthermore, each coefficient in the linear combination is bounded by  $\Delta$  in absolute value, thus  $\langle Q_i, z \rangle$  is itself Gaussian with mean 0 and variance  $\sigma^2 \leq t_Q \Delta^2$ . By standard tail bounds for Gaussian distributions, we get that  $\Pr[|\langle Q_i, z \rangle| > \alpha] = e^{-O(\alpha^2/\sigma^2)}$ . Setting  $\alpha = \Omega(\Delta \sqrt{t_Q \lg m}) = \Omega(\sigma \sqrt{\lg m})$  this is less than  $1/m^2$ . By a union bound over all  $m$  constraints in the definition of

$K_\alpha$ , we conclude that  $\Pr[z \notin K_\alpha] < 1/m$ , i.e.  $\gamma_p(K_\alpha) > 1 - 1/m > 1/2$ . ■

We are now ready to define a sequence of vectors in  $cB_2^D$  and apply Theorem 3. Letting  $D_j$  denote the  $j$ 'th column vector of  $D$ , we define the vectors  $d_1, \dots, d_n$ , where  $d_j = c/(\Delta\sqrt{t_D}) \cdot D_j$ , and  $c > 0$  is the constant in Theorem 3. Since each column of  $D$  has at most  $t_D$  non-zero entries, each bounded by  $\Delta$  in absolute value, we get that  $\|D_j\|_2 \leq \sqrt{t_D}\Delta$  for all  $j$ , and thus  $d_1, \dots, d_n \in cB_2^D$ .

Letting  $\alpha = \Theta(\Delta\sqrt{t_Q \lg m})$ , we get by Theorem 3, that there exist signs  $\varepsilon_1, \dots, \varepsilon_n \in \{-1, +1\}$  such that  $\sum_{j=1}^n \varepsilon_j d_j \in K_\alpha$ . Now define the vector  $x = (\varepsilon_1, \dots, \varepsilon_n)$ . We claim that  $\|QDx\|_\infty = O(\Delta^2 \sqrt{t_D t_Q \lg m})$ . To see this, note that  $Dx = \sum_{j=1}^n \varepsilon_j D_j = c^{-1} \Delta \sqrt{t_D} \sum_{j=1}^n \varepsilon_j d_j$ . Now consider the  $i$ 'th coordinate of  $QDx$ . This coordinate is given by the inner product  $\langle Q_i, Dx \rangle = c^{-1} \Delta \sqrt{t_D} \langle Q_i, \sum_{j=1}^n \varepsilon_j d_j \rangle$ . But since  $\sum_{j=1}^n \varepsilon_j d_j \in K_\alpha$ , this is by definition of  $K_\alpha$  bounded in absolute value by  $c^{-1} \Delta \sqrt{t_D} \alpha = O(\Delta^2 \sqrt{t_D t_Q \lg m})$ . This concludes the proof of Theorem 2.

#### 4. IMPLICATIONS

Having established Theorem 1, we now get to the fun part, all of the immediate implications. To not waste space on repeating ourselves, we refer the reader to the list of our results presented in Section 1.3 for an overview of the range searching lower bounds achieved (they follow directly by plugging in the discrepancy lower bounds from Section 1.2 in Theorem 1). Thus for the remainder of the section, we present our combinatorial discrepancy upper bound for orthogonal range searching (axis-aligned rectangles).

##### 4.1. Combinatorial Discrepancy Upper Bounds

In this section we review the classic data structure solution to orthogonal range searching [8]. We give a rather thorough review to also make clear the translation of a data structure into matrix factorization. We finally summarize the implications of combining the solution with Theorem 1.

*1-d Orthogonal Range Searching:* We set out in the one-dimensional case. Here the input to orthogonal range searching is a set  $P$  of  $n$  points on the real line, and the goal is to support computing the group sum of the weights assigned to the input points intersecting a query interval. This problem clearly includes the partial sums problem as a special case.

The solution to this problem, is to construct a complete binary tree  $\mathcal{T}$  over the input points ordered by their coordinates. Each leaf of  $\mathcal{T}$  is associated to one input point, and each internal node  $v$  is associated to the range of points associated to the leaves of the subtree rooted at  $v$ . The data structure stores one group sum for each node in  $\mathcal{T}$ . The group sum stored for a node  $v$  is simply the sum of the weights assigned to the input points associated to  $v$ .

Let  $[x_1, x_2]$  be a range query. To answer the range query, we first find the two leaves  $v_1$  and  $v_2$  containing the successor of  $x_1$  and the predecessor of  $x_2$  respectively. Let  $u$  denote the lowest common ancestor of  $v_1$  and  $v_2$ . We now traverse the path from the left child of  $u$  to  $v_1$ , and for each node  $w$  that is a right child of a node on this path, but not itself on the path, we add up the group sum stored at  $w$ . We then do the same with  $v_1$  replaced by  $v_2$  and the roles of left and right reversed, and finally we add up the group sums stored at  $v_1$  and  $v_2$ . This is easily seen to sum precisely the weights of the points with a coordinate in the range  $[x_1, x_2]$ .

Since the height of  $\mathcal{T}$  is  $O(\lg n)$ , we get that the data structure answers queries in  $t_q = O(\lg n)$  group operations. The weight of each input point  $p$  is associated to one node at each level of the tree, namely the ancestor nodes of the leaf associated to  $p$ . Thus the update time is also  $t_u = O(\lg n)$ , since an update consists of re-evaluating the group sums stored in these nodes.

For completeness, we also sketch what the matrices  $Q_{P,\mathcal{R}}$  and  $D_{P,\mathcal{R}}$  looks like for this data structure. The data matrix  $D_{P,\mathcal{R}}$  has one row for each node in  $\mathcal{T}$  and one column for each input point. The row corresponding to a node  $v$  has a 1 in the column corresponding to a point  $p$  if  $p$  is associated to  $v$ , and a 0 otherwise. The query matrix  $Q_{P,\mathcal{R}}$  has a column for each node in  $\mathcal{T}$  (i.e. for each stored group sum). Furthermore, if  $p_1, \dots, p_n$  denotes the input points ordered by their coordinate, then  $Q_{P,\mathcal{R}}$  has one row for every pair of points  $p_i$  and  $p_j$ , ( $i \leq j$ ). For the row corresponding to a pair  $p_i$  and  $p_j$ , let  $[x_i, x_j]$  denote a query range containing precisely the coordinates of the points  $p_i, \dots, p_j$ . Then that row has a 1 in each column corresponding to a node for which the stored group sum is added up when executing the above query algorithm on  $[x_i, x_j]$ , and a 0 elsewhere.

*Higher Dimensions:* The above data structure is easily generalized to higher dimensions. Let  $P$  denote the input points, and construct the one-dimensional layout on the last coordinate of the points, i.e. construct a complete binary tree over the sorted list of points. Each node in the tree no longer maintains the group sum of the weights assigned to points in the subtree, but instead it stores a  $(d-1)$ -dimensional data structure on the projection of the points in the subtree onto the first  $(d-1)$  dimensions.

A query  $[x_1, x_2] \times \dots \times [x_{2d-1}, x_{2d}]$  is answered analogous to the one-dimensional approach, except that whenever the one-dimensional data structure adds up the group sum stored in a node, we instead project the query range onto the first  $d-1$  dimensions, and ask the resulting query to the  $(d-1)$ -dimensional data structure stored in that node. Since each queried  $(d-1)$ -dimensional data structure is implemented only on points with a  $d$ 'th coordinate inside  $[x_{2d-1}, x_{2d}]$ , this correctly computes the answer to the query range.

It is easily seen that the weight of a point is included in  $O(\lg^d n)$  stored group sums, thus the update time of this data structure is  $t_u = O(\lg^d n)$ . The query algorithm

similarly adds up  $t_q = O(\lg^d n)$  stored group sums. Finally, we observe that this data structure has multiplicity 1 since the corresponding matrix factorizations use only coefficients amongst  $\{0, 1\}$ . By contraposition of Theorem 1 we thus conclude

**Corollary 1.** *For any set  $P$  of  $n$  points in  $d$ -dimensional space, it holds that  $\text{disc}(P, \mathcal{B}^d) = O(\lg^{d+1/2} n)$ , where  $\mathcal{B}^d$  denotes the set of all axis-aligned rectangles in  $\mathbb{R}^d$ .*

## 5. CONCLUSION

In this paper we established a powerful theorem relating the update and query time of dynamic range searching data structures in the group model and the combinatorial discrepancy of the corresponding range searching problem. Our result immediately implied a whole range of data structure lower bounds, and also an improved upper bound for the discrepancy of axis-aligned rectangles in dimensions  $d \geq 3$ .

We believe our result is a big leap in the right direction, but there are still a number of open problems to consider. Most importantly, we would like to remove the dependence on the multiplicity of data structures. Proving lower bounds independent of the multiplicity seems closely related to matrix rigidity, and thus might turn out to be very challenging. A breakthrough in this direction might also help towards establishing higher lower bounds for static range searching problems. On the other hand, it would also be interesting to find an example of a range searching problem for which high multiplicity helps. If possible, this seems to involve finding a completely new approach to designing data structures, and might inspire improved solutions to many natural problems. Extending the achieved lower bounds to weakly oblivious data structures would also be a major result, especially if this could be done independent of the multiplicity. Previous such lower bounds could even be extended to the cell-probe model. Obtaining a lower bound for halfspace range searching that (essentially) matches the entire tradeoff curve of [19] would also be a great result, but it seems to require many new ideas.

## ACKNOWLEDGMENT

The author wishes to thank Timothy M. Chan, Peter Bro Miltersen, Jeff M. Phillips and Elad Verbin for much useful discussion on the contents and writing of this paper.

## REFERENCES

- [1] R. Alexander, “Geometric methods in the study of irregularities of distribution,” *Combinatorica*, vol. 10, no. 2, pp. 115–136, 1990.
- [2] W. Banaszczyk, “Balancing vectors and gaussian measures of  $n$ -dimensional convex bodies,” *Random Structures & Algorithms*, vol. 12, pp. 351–360, July 1998.
- [3] N. Bansal, “Constructive algorithms for discrepancy minimization,” in *Proc. 51st IEEE Symposium on Foundations of Computer Science*, 2010, pp. 3–10.
- [4] J. Beck, “Balanced two-colorings of finite sets in the square I,” *Combinatorica*, vol. 1, no. 4, pp. 327–335, 1981.
- [5] —, “On irregularities of point sets in the unit square,” in *Combinatorics. Proc. 7th Hungarian colloquium*, 1988, pp. 63–74.
- [6] J. Beck and T. Fiala, “Integer-making theorems,” *Discrete Applied Mathematics*, vol. 3, pp. 1–8, February 1981.
- [7] D. Bednarchak and M. Helm, “A note on the Beck-Fiala theorem,” *Combinatorica*, vol. 17, no. 1, pp. 147–149, 1997.
- [8] J. L. Bentley, “Multidimensional divide-and-conquer,” *Communications of the ACM*, vol. 23, no. 4, pp. 214–229, 1980.
- [9] D. Bilyk, M. T. Lacey, and A. Vagharshakyan, “On the small ball inequality in all dimensions,” *Journal of Functional Analysis*, vol. 254, pp. 2470–2502, May 2008.
- [10] T. M. Chan, “Optimal partition trees,” in *Proc. 26th ACM Symposium on Computational Geometry*, 2010, pp. 1–10.
- [11] B. Chazelle, “Lower bounds for off-line range searching,” in *Proc. 27th ACM Symposium on Theory of Computation*, 1995, pp. 733–740.
- [12] —, “A spectral approach to lower bounds with applications to geometric searching,” *SIAM Journal on Computing*, vol. 27, pp. 545–556, 1998.
- [13] —, *The Discrepancy Method: Randomness and Complexity*. Cambridge University Press, 2000.
- [14] B. Chazelle and A. Lvov, “A trace bound for the hereditary discrepancy,” in *Proc. 16th ACM Symposium on Computational Geometry*, 2000, pp. 64–69.
- [15] M. L. Fredman and M. Saks, “The cell probe complexity of dynamic data structures,” in *Proc. 21st ACM Symposium on Theory of Computation*, 1989, pp. 345–354.
- [16] M. L. Fredman, “The complexity of maintaining an array and computing its partial sums,” *Journal of the ACM*, vol. 29, pp. 250–260, January 1982.
- [17] J. Matoušek, “Tight upper bounds for the discrepancy of half-spaces,” *Discrete and Computational Geometry*, vol. 13, pp. 593–601, 1995.
- [18] —, *Geometric Discrepancy*. Springer, 1999.
- [19] J. Matoušek, “Efficient partition trees,” *Discrete and Computational Geometry*, vol. 8, pp. 315–334, 1992.
- [20] M. Pătraşcu, “Lower bounds for 2-dimensional range counting,” in *Proc. 39th ACM Symposium on Theory of Computation*, 2007, pp. 40–46.
- [21] M. Pătraşcu and E. D. Demaine, “Logarithmic lower bounds in the cell-probe model,” *SIAM Journal on Computing*, vol. 35, pp. 932–963, April 2006.
- [22] K. F. Roth, “Remark concerning integer sequences,” *Acta Arithmetica*, vol. 9, pp. 257–260, 1964.
- [23] A. Srinivasan, “Improving the discrepancy bound for sparse matrices: better approximations for sparse lattice approximation problems,” in *Proc. 8th ACM/SIAM Symposium on Discrete Algorithms*, 1997, pp. 692–701.