The Bichromatic Rectangle Problem in High Dimensions

Jonathan Backer[*] J. Mark Keil[†]

Abstract

Given a set of blue points and a set of red points in d-dimensional space, we show how to find an axis-aligned hyperrectangle that contains no red points and as many blue points as possible. Our algorithm enumerates the set of *relevant* hyperrectangles (inclusion maximal axis-aligned hyperrectangles that do not contain a red point) and counts the number of blue points in each one. The runtime of our algorithm depends on the total number of relevant hyperrectangles. We prove asymptotically tight bounds on this quantity in the worst case. The techniques developed directly apply to the maximum empty rectangle problem in high dimensions.

1 Introduction

We show how to solve instances of the bichromatic shape problem: find a figure of a certain shape that contains no red points and as many blue points as possible. Specifically, we present an efficient solution to the bichromatic axis-aligned hyperrectangle problem. The restriction to axis-aligned shapes is common in the literature. As is customary, we assume that hyperrectangles are axis-aligned, unless otherwise noted, to save space. Two other common conventions that we follow are the use of d to denote dimension and n to denote the total number of red and blue points.

Recently, Aronov and Har-Peled posed the bichromatic problem for a variety of shapes [2], including squares and rectangles. In their paper, they present an $(1+\varepsilon)$-approximation algorithm for the bichromatic ball problem that runs in $O(n^{\lceil d/2 \rceil}(\varepsilon^{-2}\log n)^{\lceil d/2+1 \rceil})$ time, for dimensions $d \geq 3$.

Eckstein et al. explore the bichromatic problem for hyperrectangles in high-dimensions, a problem motivated by data-analysis [7]. They show that the bichromatic hyperrectangle problem is NP-hard, for arbitrarily high dimensions. They also present a simple $O(n^{2d+1})$ time algorithm, for any fixed dimension d. As this runtime grows exponentially in d, they develop a heuristic algorithm suitable for high dimensions. In a subsequent paper, Liu and Nediak propose a $O(n^2 \log n)$ time and $O(n)$ space algorithm for the two-dimensional bichromatic rectangle problem [10].

The bichromatic shape problem is a natural variant of the maximum empty shape problem: find a figure of a certain shape that is contained in a given bounding box, contains no red points, and has as large a volume as possible. The techniques used to solve these two different problems are quite similar. The enumeration approach used in this paper was previously applied to the maximum empty rectangle problem in two and three dimensions [11, 13, 12, 5]. In a separate paper [3], we solve the two-dimensional bichromatic square and rectangle problems in $O(n \log n)$ and $O(n \log^3 n)$ space respectively. Our approach uses techniques previously applied to the maximum empty square and rectangle problems [4, 1]. All of the techniques just mentioned require at least linear storage, which makes them unsuitable for large data sets. Edmonds et al. describe a heuristic solution to the maximum empty rectangle problem that typically requires much less space [8].

For completeness, we also mention the maximum discrepancy problem, which is similar to the bichromatic shape problem: find a figure of a certain shape that maximises the difference between the number of contained blue points and number of contained red points. Dobkin et al. solve the maximum discrepancy problem for rectangles in $O(n^2 \log n)$ time [6]. In their paper, they relate various discrepancy problems to problems in machine learning and computer graphics.

2 Preliminaries

By hyperrectangle, we mean a Cartesian product of d closed intervals $\prod_{i=1}^{d}[a_i, b_i]$. Let E be any hyperrectangle *enclosing* all of the red and blue points. We say that a hyperrectangle is feasible, if it is contained in E and no red point lies in its interior. To solve the bichromatic hyperrectangle problem, we only need to consider feasible hyperrectangles. By relevant hyperrectangle (RHR), we mean a feasible hyperrectangle that is not *properly* contained in any feasible hyperrectangle. Clearly, some RHR solves the bichromatic hyperrectangle problem.

A side of a RHR is a region of its boundary most extreme in some co-ordinate axis (i.e. boundary points such that the ith co-ordinate equals a_i or b_i). Clearly, there are $2d$ sides to an RHR. We say that a side is *supported*, if its interior touches a red point or E. A property of RHRs that we repeatedly exploit is that

[*]Department of Computer Science, University of Saskatchewan, jonathan.backer@usask.ca

[†]Department of Computer Science, University of Saskatchewan, keil@cs.usask.ca

each side of a RHR is supported: if some side S is unsupported, we can push S out while keeping the RHR feasible, contradicting that the RHR is maximal.

In Section 3, we bound the number of RHRs in the worst case. This complements a prior bound of $O(n \log^{d-1} n)$ RHRs on average [5], under modest assumptions on the input (essentially that the red points are chosen independently).

In Section 4, we show how to enumerate all RHRs and count the number of blue points contained in each one. Such enumeration can also be used to solve the maximum empty rectangle problem. In three dimensions, our enumeration is asymptotically slower by a $\log n$ factor than the algorithm by Datta and Soundaralakshmi [5], in the worst case. However, our enumeration is much simpler, it has the same asymptotic average case run time, and it generalises to higher dimensions.

3 Number of Relevant Rectangles

We now prove the following result conjectured in [5].

Theorem 1 *For any fixed dimension d, there are $\Theta(n^d)$ RHRs, in the worst case.*

3.1 Worst-case Lower Bound

In this section, we arrange n red points so that there are at least $(n/d)^d$ RHRs that contain the origin. This result is a generalisation of a 3D construction [5]. As an enclosing hyperrectangle E, we take the width two hypercube centred at the origin.

Let x_i^+ be the unit vector in the positive direction of the i^{th} co-ordinate axis, for $1 \le i \le d$. Define x_i^- analogously. We first partition these $2d$ vectors into d pairs so that if x_i^+ is paired with either x_j^+ or x_j^-, then x_i^- is paired with neither x_j^+ nor x_j^-. Figure 1 illustrates the odd and even cases of one pairing scheme.

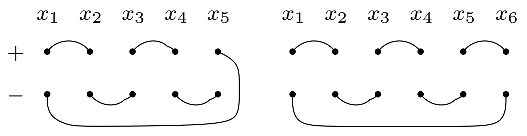
x_1 x_2 x_3 x_4 x_5
x_1 x_2 x_3 x_4 x_5 x_6
+
−

Figure 1: Partitions for $d = 5$ and $d = 6$.

For each pair of vectors α and β contained in a partition, we evenly place $\lfloor n/d \rfloor$ red points along the line segment between α and β. In the projection onto the plane P defined by α and β, most of the red points are placed at the origin, $\lfloor n/d \rfloor$ points are placed on the $-\alpha$ axis, and $\lfloor n/d \rfloor$ points are placed on the $-\beta$ axis (see Figure 2).

Let H be any RHR in $\mathbb{R}^d$ that contains the origin. To visualise H, consider its projection H' into P (see Figure 2). Let T denote the side of H furthest in direction α. The top side T' of H' is the projection of T into P. If we were to push T in or out, T' would slide up or down. Moreover, H would remain feasible, as long as T' did pass through a red point. Finally, T would be supported if and only if T' were supported (assuming H still contained the origin). Similar observations hold with respect to side R of H furthest in direction β and the right side of H'. Figure 2 illustrates the projection of $\lfloor n/d \rfloor + 1$ feasible, supported placements of T and R. These placements do not affect the placement of any of the other sides of H. Therefore, there are at least $(\lfloor n/d \rfloor + 1)^d$ RHRs containing the origin.

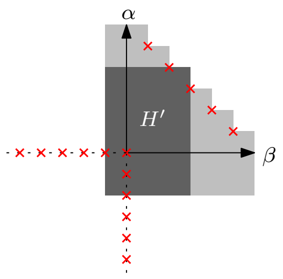

Figure 2: Potential upper right corners of H' form a staircase.

3.2 Worst-case Upper Bound

In this section, we prove that there are $O(n^d)$ RHRs by induction on d. The base case where $d = 1$ is trivial.

First, we bound the number of RHRs that are supported by E on at least one side. Let H be a RHR, and let the top and bottom of H refer to its extreme sides in the d^{th} dimension. Suppose that the top of H touches E (the other cases are similar). Let R be the region between the two hyperplanes coinciding with the top and bottom of H. To apply our inductive hypothesis, we project everything contained in R into $d-1$ dimensions (drop the d^{th} co-ordinate). Note that the projection of H is a $(d-1)$-dimensional RHR. Hence, there are $O(n^{d-1})$ RHRs supported by E on the top, for each of the $O(n)$ possible bottoms.

Second, we bound the number of RHRs that are supported by red points on each side. We do this by (a) mapping each such RHR to one of $O(n^d)$ kernels and (b) arguing that at most 4^d RHRs map to a single kernel. Our argument is simplified by assuming that the co-ordinates of the red points in each axis are unique. It is possible to symbolically perturb the set of red points so that this condition holds.

To define our mapping, we choose an arbitrary order of the $2d$ sides of a d-dimensional hyperrectangle (see Figure 3). Given a RHR H, we visit each side of H in order. If a side contains a red point on its interior, we

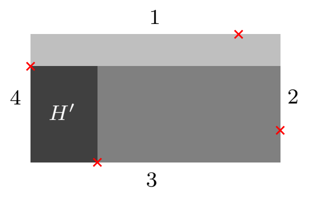

Figure 3: Deflation of a RHR in the given order the sides.

push it in until a point occurs on its boundary. When this process finishes, we are left with a *kernel* H', where every side touches a red point, but only on that side's boundary. Let P be the set of red points touching H'. Then $|P| \leq d$ because (a) each side of H touches at most one red point (by uniqueness of point co-ordinates) and (b) each point of P touches at least two sides. Hence, there are $O(n^d)$ possible H' because H' is the smallest hyperrectangle containing P and $|P| \leq d$.

Given H', we can recover H by visiting the sides of H' in reverse order and pushing each side out until its interior touches a red point. This requires knowing which sides to inflate. There are 2^{2d} such possibilities.

This counting argument was inspired by an argument applied to a related problem [9].

4 Enumerating Relevant Hyperrectangles

In this section we prove the following theorem.

Theorem 2 *For any fixed dimension d, we can solve bichromatic rectangle problem in $O(k \log^{d-2} n)$ time and $O(k+n \log^{d-2} n)$ space, where k is the number of RHRs.*

The core of our approach is an efficient enumeration of all RHRs. The details are simplified by assuming that the co-ordinates of the red points in each dimension are unique. This can be imposed with a symbolic perturbation of the point set.

To find RHRs, we sweep $\mathbb{R}^d$ from $-\infty$ to ∞ with a hyperplane P that is orthogonal to the d^{th} axis. We refer to the sides of P that extend towards ∞ and $-\infty$ in the d^{th} axis as above and below respectively. Similarly, we refer to the sides of a hyperrectangle that are parallel to P and closest to ∞ and $-\infty$ as top and bottom respectively.

Let E' be the region of the bounding hyperrectangle E that lies below P. As we sweep, we ensure that we have discovered all of the RHRs with respect to E' (i.e. the RHRs that result from taking E' as the bounding hyperrectangle and restricting our attention to the red points inside of E'). To do this, we maintain a list L of all of the RHRs with respect to E' that touch P. We represent each such RHR by how each of its sides is supported (i.e. by which red point or side of E'). Occasionally, we must add to and delete from L. All such events occur when P passes through a red point.

Let $r_1, r_2, \ldots$ denote the red points sorted in increasing order of d^{th} co-ordinate. We now describe how to update L as P passes through r_i. Let P^+ and P^- be sweep planes lying just above and below r_i respectively (see Figure 4). Let L^+ and L^- be the lists associated with P^+ and P^- respectively. The RHRs common to both L^- and L^+ are not supported by r_i on any boundary.

Let D_i be the set of RHRs in L^- such that r_i lies directly above the top of each one. By our assumption of co-ordinate uniqueness, r_i supports the top of each RHR in D_i when the sweep plane reaches r_i. So D_i is the set of RHRs deleted from L^+. To compute D_i, we perform an orthogonal range query whenever a new RHR H is discovered. Specifically, we add H to D_j, where r_j is the red point with the lowest index that lies directly above the bottom of H. If there is no such r_j, we add H to a special set D_∞. We can locate r_j in $O(\log^{d-1} n)$ time by using a range tree with fractional cascading. Such a tree requires $O(n \log^{d-1} n)$ time and space to construct.

Let A_i be the set of RHRs in L^+ that are supported on some side by r_i. Clearly, A_i is the set of RHRs added to L^+. Let H_a be a RHR of A_i. Then r_i does not support the top of H_a, which is supported by P^+. If r_i supports the bottom of H_a, then H_a is the region of E above r_i and below P^+. Otherwise, H_a corresponds to some RHR of D_i as follows (see Figure 4): Let H' be the result of pushing the top of H_a down so that it coincides with P^-. When the sweep plane is at P^-, the top of H' is supported by the plane, but one other side is not. There is at most one such unsupported side because the co-ordinates of the red points are unique. Push this one side out until it hits an obstacle. This results in a RHR H_d that belongs to D_i.

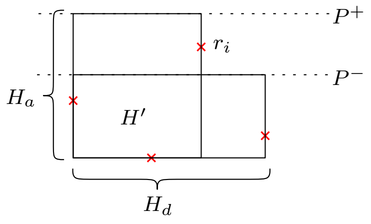

Figure 4: An added hyperrectangle H_a corresponds to a deleted hyperrectangle H_d (illustrated in 2D).

This correspondence between RHRs of A_i and RHRs of D_i suggests a procedure for generating A_i from D_i. Let H_d be a RHR of D_i. Let S be one of the $(d-1)$ axis-aligned hyperplanes through r_i that is perpendicular to P. For each such S, split H_d along S, which results in two pieces. For each piece, check if it is properly supported on all all but two sides: the side coincident with S (which will be supported by r_i) and the side coincident with P^- (which will be supported by P^+). If a piece is properly supported add it to A_i. Otherwise, discard it. This process can be executed in $O(|D_i|)$ time, for any fixed d.

The set of all RHRs are $\bigcup_i D_i$. We can count the number of blue points contained in each one using a range tree with fractional cascading. Note that we do not need to compute L explicitly (it can be constructed from the D_i). We only used it to simplify the description of our algorithm. Let k be the number of RHRs. Clearly, $n \in O(k)$ because each red point supports the bottom of at least one RHR. It is straightforward to verify that this sweep-plane algorithm takes $O(k \log^{d-1} n)$ time and $O(k + n \log^{d-1} n)$ space.

This sweep plane algorithm is similar to an algorithm for finding maximal negative orthants [9].

4.1 Saving a Logarithmic Factor

A simple observation allows us to shave a dimension off of our red-point range tree: When we query the red-point range tree for a point lying above the bottom of a RHR, we know that the desired point r_j lies above the sweep plane. Hence, the height of the bottom is irrelevant, if we remove red points from the range tree as the sweep plane passes over them.

Likewise, a simple observation allows us to shave a dimension off of our blue-point range tree: The number of points in a RHR is the number of blue points directly above the bottom minus the number of blue points directly above the top. This allows us to perform a separate sweep to count the number of blue points in each RHR. Similarly, we remove points from the blue-point range tree as the sweep plane passes over them.

With these slight modifications, our algorithm takes $O(k \log^{d-2} n)$ time and $O(k + n \log^{d-2} n)$ space.

5 Open Problems

In 2D, the bichromatic rectangle and maximum empty rectangle problems can be solved without examining all of the RHRs. In the worst case, these algorithms perform much better asymptotically than the enumeration algorithms. At the core of these non-enumeration algorithms is a fundamental observation that does not seem to generalise to higher dimensions. Still, the gap between worst case run times in 2D suggests that we can do better than enumeration in three or higher dimensions. This remains an outstanding open problem.

References

[1] A. Aggarwal and S. Suri. Fast algorithms for computing the largest empty rectangle. In *Proceedings of the third annual Symposium on Computational Geometry*, pages 278–290. ACM New York, NY, USA, 1987.

[2] B. Aronov and S. Har-Peled. On Approximating the Depth and Related Problems. *SIAM Journal on Computing*, 38(3):899–921, 2008.

[3] J. Backer and J. Keil. The bichromatic square and rectangle problems. Technical Report 2009-01, University of Saskatchewan, 2009.

[4] B. Chazelle, R. Drysdale, and D. Lee. Computing the largest empty rectangle. *SIAM Journal on Computing*, 15:300, 1986.

[5] A. Datta and S. Soundaralakshmi. An efficient algorithm for computing the maximum empty rectangle in three dimensions. *Information Sciences*, 128(1-2):43–65, 2000.

[6] D. Dobkin, D. Gunopulos, and W. Maass. Computing the maximum bichromatic discrepancy, with applications to computer graphics and machine learning. *Journal of Computer and System Sciences*, 52(3):453–470, 1996.

[7] J. Eckstein, P. Hammer, Y. Liu, M. Nediak, and B. Simeone. The maximum box problem and its application to data analysis. *Computational Optimization and Applications*, 23(3):285–298, 2002.

[8] J. Edmonds, J. Gryz, D. Liang, and R. Miller. Mining for empty spaces in large data sets. *Theoretical Computer Science*, 296(3):435–452, 2003.

[9] H. Kaplan, N. Rubin, M. Sharir, and E. Verbin. Counting colors in boxes. In *Proceedings of the eighteenth annual ACM-SIAM Symposium on Discrete algorithms*, pages 785–794. Society for Industrial and Applied Mathematics Philadelphia, PA, USA, 2007.

[10] Y. Liu and M. Nediak. Planar case of the maximum box and related problems. In *Proceeding of the Canadian Conference on Computational Geometry*, pages 11–13, 2003.

[11] A. Naamad, D. Lee, and W. Hsu. Maximum empty rectangle problem. *Discrete Appl. Math.*, 8(3):267–277, 1984.

[12] S. Nandy and B. Bhattacharya. Maximal empty cuboids among points and blocks. *Computers and Mathematics with Applications*, 36(3):11–20, 1998.

[13] M. Orlowski. A new algorithm for the largest empty rectangle problem. *Algorithmica*, 5(1):65–73, 1990.