Information Processing Letters 114 (2014) 437–445

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

# Maximum-weight planar boxes in $O(n^2)$ time (and better) ☆

Jérémy Barbay [a,1], Timothy M. Chan [b], Gonzalo Navarro [a,1], Pablo Pérez-Lantero [c,*,1,2]

[a] Department of Computer Science, University of Chile, Chile
[b] Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada
[c] Escuela de Ingeniería Civil en Informática, Universidad de Valparaíso, Valparaiso, Chile

ARTICLE INFO

*Article history:*
Received 12 November 2013
Accepted 17 March 2014
Available online 20 March 2014
Communicated by M. Yamashita

*Keywords:*
Computational geometry
Maximum box
Divide–summarize-and-conquer
Adaptive algorithms
Klee's Measure problem

ABSTRACT

Given a set $P$ of $n$ points in $\mathbb{R}^d$, where each point $p$ of $P$ is associated with a weight $w(p)$ (positive or negative), the Maximum-Weight Box problem is to find an axis-aligned box $B$ maximizing $\sum_{p\in B\cap P} w(p)$. We describe algorithms for this problem in two dimensions that run in the worst case in $O(n^2)$ time, and much less on more specific classes of instances. In particular, these results imply similar ones for the Maximum Bichromatic Discrepancy Box problem. These improve by a factor of $\Theta(\lg n)$ on the previously known worst-case complexity for these problems, $O(n^2 \lg n)$ (Cortés et al., 2009 [9]; Dobkin et al., 1996 [10]). Although the $O(n^2)$ result can be deduced from new results on Klee's Measure problem (Chan, 2013 [7]), it is a more direct and simplified (non-trivial) solution. We exploit the connection with Klee's Measure problem to further show that (1) the Maximum-Weight Box problem can be solved in $O(n^d)$ time for any constant $d \geqslant 2$; (2) if the weights are integers bounded by $O(1)$ in absolute values, or weights are $+1$ and $-\infty$ (as in the Maximum Bichromatic Discrepancy Box problem), the Maximum-Weight Box problem can be solved in $O((n^d/\lg^d n)(\lg\lg n)^{O(1)})$ time; (3) it is unlikely that the Maximum-Weight Box problem can be solved in less than $n^{d/2}$ time (ignoring logarithmic factors) with current knowledge about Klee's Measure problem.



## 1. Introduction

Consider a set $P$ of $n$ points in $\mathbb{R}^d$, such that the points are in general position (i.e., no pair of points share the same $x$ or $y$ coordinate). Each point $p$ of $P$ is assigned a weight $w(p) \in \mathbb{R}$ that can be either positive or negative. For any subset $B \subseteq \mathbb{R}^d$ let $W(B) := \sum_{p\in B\cap P} w(p)$. A *box* is an axis-aligned hyper-rectangle, and we say that the *weight* of a box $B$ is $W(B)$. We consider the Maximum-Weight Box problem, which given $P$ and $w()$ is to find a box $B$ with maximum weight $W(B)$.

Depending on the choice of the weights $w()$, this geometric optimization problem has various practical applications, such as machine learning [10] and data classification and clustering [11].

*Related work* In one dimension, the coordinates of the points matter only in the order they induce on their weights, and the problem reduces to the Maximum-Sum Consecutive Subsequence problem [4], which can be solved in $O(n)$ time if the coordinates are already sorted.

☆ A previous version of this paper appeared in the Proceedings of the 25th Canadian Conference on Computational Geometry (CCCG'13).

* Corresponding author.
*E-mail addresses:* jbarbay@dcc.uchile.cl (J. Barbay), tmchan@cs.uwaterloo.ca (T.M. Chan), gnavarro@dcc.uchile.cl (G. Navarro), pablo.perez@uv.cl (P. Pérez-Lantero).

[1] Partially funded by Millennium Nucleus Information and Coordination in Networks ICM/FIC P10-024F, Mideplan, Chile.
[2] Partially supported by grant CONICYT, FONDECYT/Iniciación 11110069, Chile.

http://dx.doi.org/10.1016/j.ipl.2014.03.007
0020-0190/

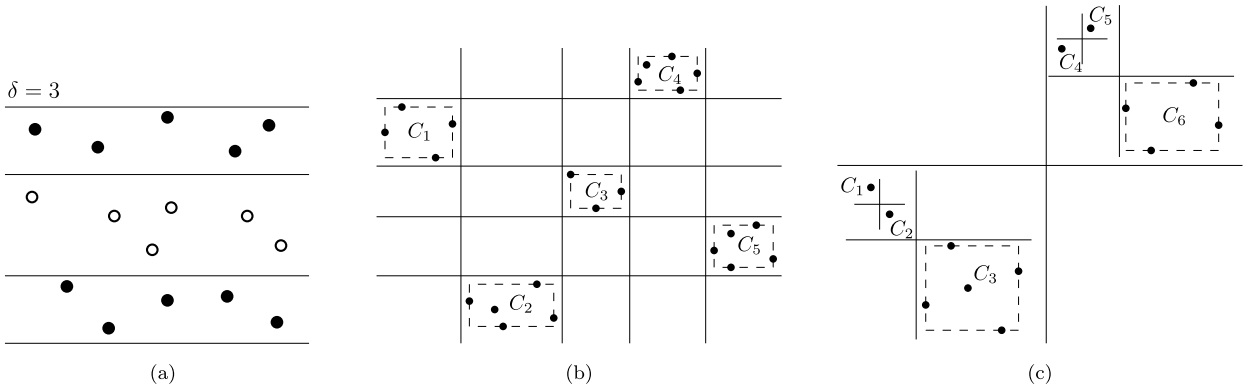


**Fig. 1.** (a) A point set composed of $\delta = 3$ strips. The points with positive weight are represented as solid dots, and the points with negative weight as tiny circles. (b) A cluster partition $(C_1, \ldots, C_5)$. (c) A diagonalization $(C_1, \ldots, C_6)$.

In 2009, Cortés et al. [9] solved the dynamic version of this problem supporting updates of weights for a fixed point set. They described a data structure called MCS-tree, which supports in $O(\lg n)$ time both updates and Maximum-Sum Consecutive Subsequence queries on any interval of the sequence of points. The Maximum-Weight Box problem in two dimensions was introduced by Cortés et al. [9], who gave an algorithm running in time within $O(n^2 \lg n)$ and space within $O(n)$. They reduce any instance of the Maximum-Weight Box problem in two dimensions to $O(n^2)$ instances of the problem in one dimension, each solved dynamically in $O(\lg n)$ with an MCS-tree.

In 2011, Bautista-Santiago et al. [3] considered convex objects other than axis-aligned rectangles, and gave an $O(n^3)$-time algorithm that finds a convex polygon maximizing the sum of the weights of the points it contains. In the case of a half-plane of maximum weight, the problem can be easily solved in $O(n^2)$ time by using duality in the plane, and it is 3SUM-hard [5]. Combining the standard lifting transformation and duality in three dimensions, the disk of maximum weight can be found in $O(n^3)$ time.

In 2012, Barbay et al. [2] generalized the Maximum-Weight Box problem to the Optimal Planar Box problem, where the sum of the weights is replaced by any monotone decomposable function $f()$, and described several adaptive improvements on Cortés et al.'s solution [9]. These include adapting to strips and other clusterings of the input, which we also consider in this work (see Sections 3 and 4). They also replaced MCS-trees by variants based on splay trees [8], which yielded an adaptive variant executing in time $O(n^2(1 + \lg(1 + \lambda/n)))$ and linear space, where $\lambda \in [0..n^2]$ is the sum of the distances between the insertion positions of the consecutive points according to their $x$-coordinate, when given in the order of their $y$-coordinate. All of their algorithms perform in time $\Theta(n^2 \lg n)$ in the worst case over all instances of $n$ points.

We consider the Maximum-Weight Box problem in two dimensions on a set $P$ of $n$ weighted points, such that no pair of points share the same $x$ or $y$ coordinate.

*Basic definitions* A *strip* is the area delimited by two lines parallel to the same axis. Given the point set $P$, we say that a strip $S$ is *monochromatic* if $S \cap P$ is not empty and the weights of all elements of $S \cap P$ have the same sign. A monochromatic strip $S$ is *positive* (resp. *negative*) if $S$ contains points of $P$ with positive (resp. negative) weights. We say that $P$ *is composed of* $\delta$ *strips* if $P$ can be covered by $\delta$ (parallel) pairwise disjoint monochromatic strips of alternating signs (see Fig. 1a). Given any bounded set $S \subset \mathbb{R}^2$, let $\mathrm{Box}(S)$ denote the smallest box covering $S$. We say that $(C_1, C_2, \ldots, C_k)$ is a *cluster partition* of $P$ if $\{C_1, C_2, \ldots, C_k\}$ is a partition of $P$ and in every axis the orthogonal projections of $\mathrm{Box}(C_1), \mathrm{Box}(C_2), \ldots, \mathrm{Box}(C_k)$ are pairwise disjoint (see Fig. 1b). A cluster partition $(C_1, C_2, \ldots, C_k)$ of $P \subset \mathbb{R}^2$ is a *diagonalization* of $P$ if (a) $k \geqslant 2$ and there is an index $j \in [1..k-1]$ such that sets $C_1 \cup \cdots \cup C_j$ and $C_{j+1} \cup \cdots \cup C_k$ belong to opposed quadrants defined by a horizontal and a vertical line, and $(C_1, \ldots, C_j)$ and $(C_{j+1}, \ldots, C_k)$ are diagonalizations of $C_1 \cup \cdots \cup C_j$ and $C_{j+1} \cup \cdots \cup C_k$, respectively, or (b) $k = 1$ and the points in $C_1$ cannot be further clustered into a diagonalization other than $(C_1)$ (see Fig. 1c).

*Results* We obtain the following results for the Maximum-Weight Box problem in two dimensions. All of our algorithms use space linear in the number of input points.

- Over instances composed of $n$ weighted points, each of our algorithms runs in $O(n^2)$ time (Theorem 2.2).
- If the point set $P$ is composed of $\delta \in [1..n]$ (either horizontal or vertical) strips, our algorithm executes adaptively in $\mathrm{SORT}(n) + O(\delta n) \subset O(n \lg n + \delta n) \subset O(n^2)$ time (Theorem 3.2), where $\mathrm{SORT}(n)$ is the time required to sort the elements of $P$ by the $x$-coordinates and by the $y$-coordinates, which is within $O(n \lg n)$ in the comparison model, and can be for instance within $O(n\sqrt{\lg \lg n}) \subset o(n \lg n)$ in the RAM model with randomization or within $O(n \lg \lg n) \subset o(n \lg n)$ deterministic, if the coordinates of the points are integer numbers [13].
- Given a cluster partition $(C_1, C_2, \ldots, C_k)$ of $P$, where cluster $C_i$ contains $n_i$ points for every $i \in [1..k]$ and is composed of $\delta_i$ strips, our algorithm runs in $O(\sum_{i=1}^{k} n_i \delta_i + k^2) \subset O(\sum_{i=1}^{k} n_i^2 + k^2) \subset O(n^2)$ time (Theorem 4.3).

- There exists a unique diagonalization $(C_1, C_2, \ldots, C_k)$ of the point set $P$ (it might be $(C_1) = (P)$ in the worst case) and our algorithm finds it in $O(n \lg n)$ time (Lemma 4.4). A maximum-weight box can be computed in overall $O(n \lg n + \sum_{i=1}^{k} n_i \delta_i) \subset O(n \lg n + \sum_{i=1}^{k} n_i^2) \subset O(n^2)$ time (Theorem 4.5), where $O(n \lg n)$ is the time to construct such a diagonalization (which is distinct from SORT$(n)$, the time required to sort the points by their coordinates, in the results mentioned above).

*Applications to other known problems* Let $P$ be a set of $n$ planar points, each being colored either red or blue.

The Maximum Bichromatic Discrepancy Box problem [9,10] is to find a box that maximizes the absolute difference between the numbers of red and blue points it contains, and was solved in $O(n^2 \lg n)$ time by Dobkin et al. [10]. Any instance of this problem can be reduced to two particular instances of the Maximum-Weight Box problem [9]. In one, red points have weight $+1$ and blue points weight $-1$, and conversely in the other. Then our results imply an $O(n^2)$ worst-case time algorithm, and adaptive algorithms as well, for the Maximum Bichromatic Discrepancy Box problem, improving upon previous $O(n^2 \lg n)$-time algorithms [9,10].

The Maximum Box problem [9,11,14] is to find a box $B$ containing the maximum number of blue points and no red point. Eckstein et al. [11] introduced it in general dimension, proving that if the dimension $d$ of the points is part of the input then the problem is NP-hard. In two dimensions it was later solved in $O(n^2 \lg n)$ time by Liu and Nediak [14]. In 2010 Backer et al. [1] showed that the Maximum Box problem in two dimensions can be solved in $O(n \lg^3 n)$ time and $O(n \lg n)$ space, and that for any fixed dimension $d \geqslant 3$ it can be solved in time within $O(n^d \lg^{d-2} n)$.

Any instance of the Maximum Box problem is equivalent to a particular case of the Maximum-Weight Box problem in which blue points have weight $+1$ and red points have weight $-\infty$ [9]. Then our techniques imply an $O(n^2)$ worst-case time algorithm for this problem, and adaptive algorithms as well. While this time complexity is worse than the best known solution [1], it requires only linear space.

Note that our specialized results are faster on some classes of instances that arise naturally in applications, such as instances where one needs to find a maximum box over an imbalanced red–blue dataset in data mining and/or data analysis [11,12,16]. Generally, if the ratio of the number of blue points over the number of red points is within $o(1)$ or $\omega(1)$,[3] then our techniques achieve $o(n^2)$ time on an instance of $n$ points.

*Higher dimensions and lower bounds* Our worst-case result in two dimensions can be seen as a particular case of recent results related to Klee's Measure problem [7], yet it is a more direct and simple solution (which we further improve on various classes of instances). By exploiting the connections between these two problems, we obtain several further results:

- We show that the Maximum-Weight Box problem can be solved in time $O(n^d)$ in any constant dimension $d \geqslant 2$. The best previous result was $O(n^{2d-2} \lg n)$ [9].
- We show that, when the weights are all $O(1)$, the Maximum-Weight Box problem can be solved in time within $O((n^d / \lg^d n)(\lg \lg n)^{O(1)})$. This improvement applies, in particular, in simpler problems such as Maximum Bichromatic Discrepancy problem and Maximum Box problem, where the best previous algorithms for the former run in time $O(n^2 \lg n)$ for $d = 2$ [9,10], and the best for the latter require time $O(n^d \lg^{d-2} n)$ for $d \geqslant 3$ [1].
- By reducing from the Weighted Depth problem, we show that the Maximum-Weight Box problem is $W[1]$-hard, and unlikely to be solved in time within $o(n^{d/2})$: such an improvement would require a breakthrough on the current knowledge of Klee's Measure problem and impact on a large set of related problems in computational geometry.

*Outline* In Section 2 we describe the general $O(n^2)$-time algorithm. In Section 3 we describe the adaptive algorithm running in SORT$(n) + O(\delta n)$ time, where SORT$(n)$ is the time required to sort the elements of $P$ by their $x$- and $y$-coordinates and $\delta$ is the number of strips of the point set. In Section 4 we present the results concerning cluster partitions and diagonalizations. Finally, in Section 5, we discuss further results in connection to Klee's Measure problem, such as extensions to higher dimensions, polylogarithmic-factor speedups, and lower bounds.

## 2. Quadratic worst-case time algorithm

Assume the elements of $P$ are sorted twice, first by $x$-coordinates and second by $y$-coordinates, in SORT$(n)$ time.

We say that $X \subseteq P$ is a *box set* if $X$ is the intersection of $P$ with some box. For any box set $X \subseteq P$ we define the *score* of $X$, $\mathtt{S}(X)$, as the following four boxes (see Fig. 2). Let $[x_1, x_2] \times [y_1, y_2] := \mathtt{Box}(X)$:

(1) $\mathtt{Box}(X)$;
(2) a box $\mathtt{B}_L(X) \subseteq \mathtt{Box}(X)$ of $X$ of maximum weight, such that it is of the form $[x_1, x] \times [y_1, y_2]$ for $x_1 \leqslant x \leqslant x_2$;
(3) a box $\mathtt{B}_R(X) \subseteq \mathtt{Box}(X)$ of $X$ of maximum weight, such that it is of the form $[x, x_2] \times [y_1, y_2]$, for $x_1 \leqslant x \leqslant x_2$; and
(4) a box $\mathtt{B}_0(X) \subseteq \mathtt{Box}(X)$ of $X$ of maximum weight, such that it is of the form $[x, x'] \times [y_1, y_2]$, for $x_1 \leqslant x \leqslant x' \leqslant x_2$.

For each of these boxes we keep only two opposed vertices defining it and its weight, so that representing a box set $X$ by $\mathtt{S}(X) := (\mathtt{Box}(X), \mathtt{B}_L(X), \mathtt{B}_R(X), \mathtt{B}_0(X))$ requires only constant space.

We say that a box set $X \subseteq P$ is *scored* if $\mathtt{S}(X)$ is computed, and we use $\mathtt{Box}(X)$ to represent $X$ instead of $X$

[3] All our asymptotic notations are for $n$ growing to infinity and other parameters, such as $\delta$, fixed.

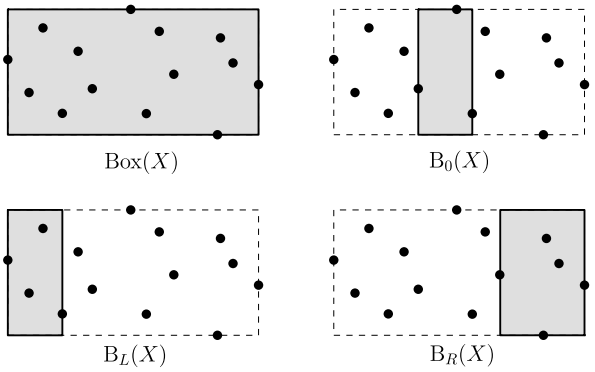


**Fig. 2.** The score $\mathrm{S}(X) = (\mathrm{Box}(X), \mathrm{B}_L(X), \mathrm{B}_R(X), \mathrm{B}_0(X))$ of a box set $X \subseteq P$.

itself. Let the operator $\oplus : 2^P \times 2^P \to 2^P$ be defined over all pairs $(X_1, X_2)$ of scored box sets of $P$ such that: $X_1$ and $X_2$ can be separated with a vertical line, $X_1$ is to the left of $X_2$, and $X_1 \cup X_2$ is a box set. Then $X_1 \oplus X_2$ returns the scored set $X_1 \cup X_2$, and it can be computed in $O(1)$ time from the next observations:

$$
\begin{aligned}
&W(X_1 \cup X_2) = W(X_1) + W(X_2)\\
&W\big(\mathrm{B}_L(X_1 \cup X_2)\big)\\
&\quad = \max\big\{W\big(\mathrm{B}_L(X_1)\big), W(X_1) + W\big(\mathrm{B}_L(X_2)\big)\big\}\\
&W\big(\mathrm{B}_R(X_1 \cup X_2)\big)\\
&\quad = \max\big\{W\big(\mathrm{B}_R(X_2)\big), W(X_2) + W\big(\mathrm{B}_R(X_1)\big)\big\}\\
&W\big(\mathrm{B}_0(X_1 \cup X_2)\big) = \max\big\{W\big(\mathrm{B}_0(X_1)\big), W\big(\mathrm{B}_0(X_2)\big),\\
&\qquad\qquad W\big(\mathrm{B}_R(X_1)\big) + W\big(\mathrm{B}_L(X_2)\big)\big\}
\end{aligned}
$$

Notice that by applying the operators $\oplus$ to singletons $\{p\}$ over all points $p$ of $P$ in left-to-right order, we can compute $\mathrm{B}_0(P)$, i.e., the maximum-weight vertical strip, in time within $O(n)$. After projection to the $x$-axis, this immediately gives a linear-time algorithm for the Maximum-Sum Consecutive Subsequence problem, studied by Bentley [4] and often taught in undergraduate algorithms classes.

Let $\mathcal{S}$ be a horizontal strip such that exactly $m$ points of $P$ are not in $\mathcal{S}$. The vertical lines passing through the $m$ points of $P \setminus \mathcal{S}$ split $\mathcal{S}$ into $m+1$ boxes denoted $\mathcal{S}_1$, $\mathcal{S}_2, \ldots, \mathcal{S}_{m+1}$ from left to right. Let $B$ be a box of maximum weight that has its top side above $\mathcal{S}$ and its bottom side below $\mathcal{S}$, and let $i, j \in [1..n+1]$ be the indices such that the left and right sides of $B$ intersect $\mathcal{S}_i$ and $\mathcal{S}_j$, respectively. If $i < j$, then $W(B \cap \mathcal{S}_i)$ and $W(B \cap \mathcal{S}_j)$ are precisely $W(\mathrm{B}_R(P \cap \mathcal{S}_i))$ and $W(\mathrm{B}_L(P \cap \mathcal{S}_j))$, respectively (see Fig. 3). Therefore we have $W(B) = W(\mathrm{B}_R(P \cap \mathcal{S}_i)) + \sum_{t=i+1}^{j-1} W(\mathcal{S}_t) + W(\mathrm{B}_L(P \cap \mathcal{S}_j)) + W(B \setminus \mathcal{S})$. On the other hand, if $i = j$, then $W(B)$ equals $W(\mathrm{B}_0(P \cap \mathcal{S}_i))$.

Consider the following Strip-Constrained Maximum-Weight Box problem: *Let $\mathcal{P}$ be a weighted point set and $\mathcal{S}$ be a horizontal strip so that: $\mathcal{P} \setminus \mathcal{S}$ consists of $n$ points already sorted from left to right; $\mathcal{S}$ splits $\mathcal{P} \setminus \mathcal{S}$ into two halves; the vertical lines through the points of $\mathcal{P} \setminus \mathcal{S}$ split $\mathcal{S}$ into the boxes $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{n+1}$ from left to right; and the points of $\mathcal{P} \cap \mathcal{S}$ are summarized by the scored box sets $\mathcal{P} \cap \mathcal{S}_1, \ldots, \mathcal{P} \cap \mathcal{S}_{n+1}$. Find a maximum-weight box of $\mathcal{P}$, with the top side above $\mathcal{S}$ and the bottom side below $\mathcal{S}$.*

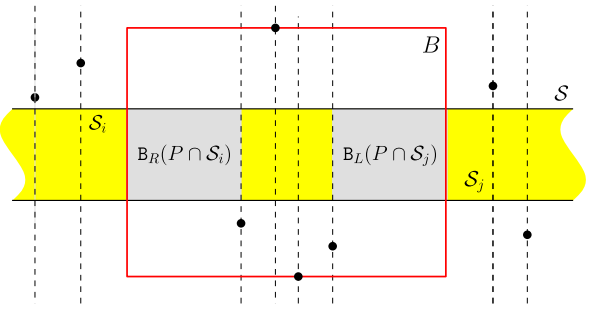


**Fig. 3.** The strip $\mathcal{S}$ is partitioned into $m+1$ boxes $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{m+1}$ by the vertical lines passing through the $m$ points in $P \setminus \mathcal{S}$. If the left and right sides of an optimal box $B$ cross $\mathcal{S}_i$ and $\mathcal{S}_j$, respectively, then they are determined by $\mathrm{B}_R(P \cap \mathcal{S}_i)$ and $\mathrm{B}_L(P \cap \mathcal{S}_j)$.

The key to our new solution is an $O(n^2)$-time algorithm for this constrained problem, using an approach which may be nick-named "divide–summarize-and-conquer".

**Lemma 2.1.** *The* Strip-Constrained Maximum-Weight Box *problem admits a solution in $O(n^2)$ time and $O(n)$ space.*

**Proof.** Let $F(n)$ denote the time required to solve a given instance of the Strip-Constrained Maximum-Weight Box problem over $n$ points. We apply divide-and-conquer: Split the points of $\mathcal{P}$ above (resp. below) $\mathcal{S}$ into two halves with a horizontal line $\ell_1$ (resp. $\ell_2$). Let $P_1$ denote the points above $\ell_1$, $P_2$ denote the points in between $\ell_1$ and $\mathcal{S}$, $P_3$ denote the points in between $\mathcal{S}$ and $\ell_2$, and $P_4$ denote the points below $\ell_2$. Then the problem can be reduced to the next four subproblems:

(1) the points of $P_1 \cup P_4$ outside a strip $\mathcal{S}'$ covering $P_2 \cup P_3 \cup \mathcal{S}$;
(2) the points of $P_1 \cup P_3$ outside a strip $\mathcal{S}'$ covering $P_2 \cup \mathcal{S}$;
(3) the points of $P_2 \cup P_3$ outside the strip $\mathcal{S}' = \mathcal{S}$; and
(4) the points of $P_2 \cup P_4$ outside a strip $\mathcal{S}'$ covering $P_3 \cup \mathcal{S}$.

The reduction to subproblem (1) can be done in time within $O(n)$ as follows: Take each point $p$ of $P_2 \cup P_3$ and compute the score $\mathrm{S}(\{p\})$. Simulate the merging of the left-to-right orders of $P_1 \cup P_4$, $P_2 \cup P_3$, and $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{n+1}$ (each of which can be obtained in $O(n)$ time) to compute the corresponding scored box sets in the new strip $\mathcal{S}'$. This computation can be done by applying the operator $\oplus$ to successive scored box sets in between consecutive points of $P_1 \cup P_4$ in the left-to-right order. The reductions to subproblems (2)–(4) are similar.

The base case occurs when $n \in \{1, 2\}$. In the most general setting ($n = 2$) we have one point $p_1$ above $S$ and one point $p_2$ below $S$, defining boxes $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ on $\mathcal{S}$. Assume w.l.o.g. that $p_1$ is to the left of $p_2$ and $w(p_1)$, $w(p_2) > 0$ (for example, if $w(p_1) < 0$, we can eliminate $p_1$). Then the solution is $\mathrm{B}_0((\mathcal{P} \cap \mathcal{S}_1) \cup \{p_1\} \cup (\mathcal{P} \cap \mathcal{S}_2) \cup \{p_2\} \cup (\mathcal{P} \cap \mathcal{S}_3))$, which can be computed in constant time

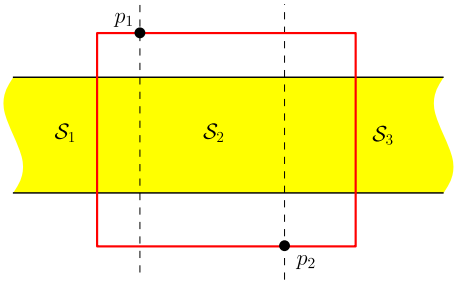


**Fig. 4.** The base case of the algorithm of Lemma 2.1.

by applying the operator $\oplus$ to the scored box sets $\mathcal{P} \cap \mathcal{S}_1$, $\{p_1\}$, $\mathcal{P} \cap \mathcal{S}_2$, $\{p_2\}$, and $\mathcal{P} \cap \mathcal{S}_3$ (see Fig. 4).

This yields the recurrence $F(n) \in 4F(n/2) + O(n)$, where $F(1) \in O(1)$. Then $F(n) \in O(n^2)$. The space $G(n)$ is within $O(n)$: the four subproblems are solved independently one after the other, and the recurrence is $G(n) \in G(n/2) + O(n)$, whose solution is within $O(n)$. □

The reduction from the original Maximum-Weight Box problem to the constrained problem follows from a more straightforward divide-and-conquer:

**Theorem 2.2.** *The* Maximum-Weight Box *problem admits a solution in $O(n^2)$ time and $O(n)$ space on instances of n points.*

**Proof.** We first sort the points of $P$ by their $x$-coordinates in SORT($n$) time and then apply a recursive procedure, whose time over $n$ weighted points will be $T(n)$. The recursion applies divide-and-conquer as follows: Draw a horizontal strip $\mathcal{S}$ (a line) splitting $P$ into two halves $P_1$ and $P_2$, where $P_1$ is above $\mathcal{S}$ and $P_2$ is below $\mathcal{S}$. Then we can find a maximum-weight box $B_1$ for $P_1$, a maximum-weight box $B_2$ for $P_2$, and a maximum-weight box $B_{1,2}$ for $P_1 \cup P_2$ restricted to intersect $\mathcal{S}$. Then the box among $B_1$, $B_2$, and $B_{1,2}$ maximizing $W()$ is the solution. To compute $B_{1,2}$ we will use the solution for the Strip-Constrained Maximum-Weight Box problem over $P$ and $\mathcal{S}$, for which we split $\mathcal{S}$ into $n+1$ empty scored boxes $\mathcal{S}_1, \ldots, \mathcal{S}_n$ according to all the $x$-coordinates of $P$. This requires $O(n)$ time and then Lemma 2.1 allows us to compute $B_{1,2}$ in $O(n^2)$ time and $O(n)$ space. Since $B_1$ and $B_2$ are computed recursively, the time complexity is $T(n) \in 2T(n/2) + O(n^2)$, where $T(1) \in O(1)$. Hence $T(n) \in O(n^2)$. As for the space $S(n)$, the three subproblems are solved independently one after the other, and thus it holds that $S(n) \in \max\{S(n/2), S(n/2), O(n)\} \subseteq O(n)$. □

## 3. $\delta$-Sensitive analysis

Assume that $P$ is composed of $\delta \in [1..n]$ strips, and suppose w.l.o.g. that these strips are horizontal. These strips can be identified in time within $O(n)$ from the sorting of the points in $P$ by their $y$-coordinates. One does not need to consider boxes whose horizontal sides are in the middle of some of these strips: there always exists an optimal box such that each horizontal side is aligned with an edge of some strip; specifically, the top (resp. bottom) of an optimal box will align with a positive point at the top (resp. bottom) of a positive strip. Using this observation we refine the results of Section 2.

**Lemma 3.1.** *The* Strip-Constrained Maximum-Weight Box *problem admits a solution in $O(\delta n)$ time and $O(n)$ space if the points of $\mathcal{P}$ above (resp. below) $\mathcal{S}$ are composed of $\delta/2$ strips.*

**Proof.** Let $F(n, \delta)$ denote the time required to solve the problem. We modify the divide-and-conquer algorithm from the proof of Lemma 2.1 as follows: We split the points above $\mathcal{S}$ with a horizontal line $\ell_1$ and the points below $\mathcal{S}$ with a horizontal line $\ell_2$, and define $P_1, \ldots, P_4$ as before. However, we choose $\ell_1$ and $\ell_2$ differently, not to ensure that each $P_i$ has $n/4$ points as in Lemma 2.1, but to ensure that each $P_i$ is composed of $\delta/4$ strips. Let $n_i$ denote the size of $P_i$ (so that $n_1 + n_2 + n_3 + n_4 = n$).

The base case arises when there is at most one strip above (resp. below) $\mathcal{S}$, and can be solved as follows: Assume w.l.o.g. that the weights of these at most two strips are positive (if one of the strips has all negative weights, we can eliminate all of its points). Then the solution is $\mathbb{B}_0(P)$, which can be computed by applying the operator $\oplus$ to the sequence, arranged in left-to-right order, consisting of $\mathcal{P} \cap \mathcal{S}_1, \ldots, \mathcal{P} \cap \mathcal{S}_{n+1}$ together with singletons $\{p_i\}$ over all $p_i$ in $\mathcal{P} \setminus \mathcal{S}$. The base case then requires $O(n)$ time.

The recurrence from Lemma 2.1 is now modified to the following:

$$F(n, \delta) \in F(n_1 + n_3, \delta/2) + F(n_1 + n_4, \delta/2) + F(n_2 + n_3, \delta/2) + F(n_2 + n_4, \delta/2) + O(n)$$

where $F(n, 1) \in O(n)$. Observe that the recursion tree for $F(n, \delta)$ has at most $\lg \delta$ levels (because $n \geqslant \delta$), and that in the $i$-th level the computation time besides recursive calls is $O(2^i n)$. Then $F(n, \delta) \in O(\delta n)$. The space is within $O(n)$ as in Theorem 2.2. □

**Theorem 3.2.** *The* Maximum-Weight Box *problem admits a solution in* SORT($n$) $+ O(\delta n)$ *time and $O(n)$ space on instances of n points composed of $\delta$ strips.*

**Proof.** Let $T(n, \delta)$ denote the time required to solve the Maximum-Weight Box problem over $n$ points composed of $\delta$ strips. We apply divide-and-conquer as in Theorem 2.2, but selecting strip $\mathcal{S}$ such that both resulting sets $P_1$ and $P_2$ are composed of $\delta/2$ strips, and $n_1$ points and $n_2$ points respectively. If there is only $\delta = 1$ strip then the solution is either empty (if the strip is negative) or all the points (if it is positive), so in the base case $T(n, 1) \in O(n)$. In the recursive case we have:

$$T(n, \delta) = T(n_1, \delta/2) + T(n_2, \delta/2) + F(n, \delta) \in T(n_1, \delta/2) + T(n_2, \delta/2) + O(\delta n).$$

The recursion tree of $T(n, \delta)$ has at most $\lg \delta$ levels and in the $i$-th level the computation time besides recursive calls is $O(\delta n/2^i)$, and thus $T(n, \delta) \in O(\delta n)$. Again, the space is $O(n)$ as before. □

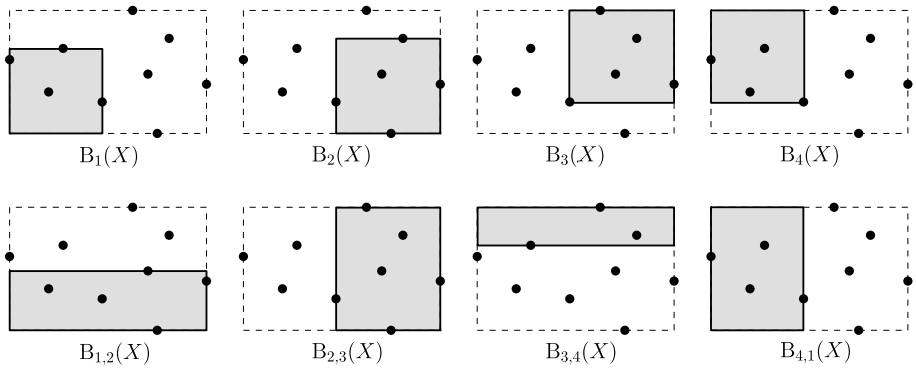


**Fig. 5.** The boxes $\mathrm{B}_1(X)$, $\mathrm{B}_2(X)$, $\mathrm{B}_3(X)$, $\mathrm{B}_4(X)$, $\mathrm{B}_{1,2}(X)$, $\mathrm{B}_{2,3}(X)$, $\mathrm{B}_{3,4}(X)$, and $\mathrm{B}_{4,1}(X)$ of a box set $X \subseteq P$.

Some naturally occurring instances will have a low number of strips. For example, instances with an unbalanced number of positive and negative points are due to contain few strips. The following corollary captures this observation.

**Corollary 3.3.** *Let $n_+$ and $n_-$ be the number of points with positive and negative weight of an instance of $n = n_+ + n_-$ points, respectively. Then the* Maximum-Weight Box *problem admits a solution in* $\mathrm{SORT}(n) + O(n \min\{n_+, n_-\})$ *time.*

**Proof.** Observe that $\delta \leqslant 2\min\{n_+, n_-\} + 1$ and apply Theorem 3.2. □

## 4. Cluster partition analysis

Let $(C_1, C_2, \ldots, C_k)$ be a cluster partition of $P$, where cluster $C_i$ contains $n_i$ points for every $i \in [1..k]$ and is composed of $\delta_i$ strips.

For any non-empty subset $X \subseteq P$ we define a set of *ten* boxes of $X$, denoted by $\mathtt{Ten}(X)$, as the set with the following maximum-weight boxes of $X$, all contained in $\mathtt{Box}(X)$: (1) $\mathtt{Box}(X)$; (2) a box of maximum weight $\mathrm{B}_{opt}(X)$ of $X$; (3)–(6) a box of maximum weight $\mathrm{B}_1(X)$ (resp. $\mathrm{B}_2(X)$, $\mathrm{B}_3(X)$, $\mathrm{B}_4(X)$) of $X$ that contains the bottom-left (resp. bottom-right, top-right, top-left) vertex of $\mathtt{Box}(X)$; and (7)–(10) a box of maximum weight $\mathrm{B}_{1,2}(X)$ (resp. $\mathrm{B}_{2,3}(X)$, $\mathrm{B}_{3,4}(X)$, $\mathrm{B}_{4,1}(X)$) of $X$ that contains the bottom (resp. right, top, left) vertices of $\mathtt{Box}(X)$ (see Fig. 5).

**Lemma 4.1.** *For any non-empty subset $X \subseteq P$ and any cluster partition $(X_1, X_2)$ of $X$, $\mathtt{Ten}(X)$ can be computed in $O(1)$ composition operations from $\mathtt{Ten}(X_1)$ and $\mathtt{Ten}(X_2)$.*

**Proof.** Suppose that cluster $X_1$ is below cluster $X_2$ (remember that $X_1$ is also to the left of $X_2$ by definition). The case in which $X_1$ is above $X_2$ is similar. The lemma follows from the next observations:

$$W(X) = W(X_1) + W(X_2)$$

$$W\big(\mathrm{B}_{opt}(X)\big) = \max\big\{W\big(\mathrm{B}_{opt}(X_1)\big), W\big(\mathrm{B}_{opt}(X_2)\big), W\big(\mathrm{B}_3(X_1)\big) + W\big(\mathrm{B}_1(X_2)\big)\big\}$$

$$W\big(\mathrm{B}_1(X)\big) = \max\big\{W\big(\mathrm{B}_1(X_1)\big), W(X_1) + W\big(\mathrm{B}_1(X_2)\big)\big\}$$

$$W\big(\mathrm{B}_2(X)\big) = \max\big\{W\big(\mathrm{B}_2(X_1)\big), W\big(\mathrm{B}_2(X_2)\big), W\big(\mathrm{B}_{2,3}(X_1)\big) + W\big(\mathrm{B}_{1,2}(X_2)\big)\big\}$$

$$W\big(\mathrm{B}_{1,2}(X)\big) = \max\big\{W\big(\mathrm{B}_{1,2}(X_1)\big), W(X_1) + W\big(\mathrm{B}_{1,2}(X_2)\big)\big\}$$

Symmetric arguments can be given for computing the weights of the other boxes. □

**Lemma 4.2.** *Given a cluster partition $(C_1, C_2, \ldots, C_k)$ of $P$ so that $\mathtt{Ten}(C_1), \mathtt{Ten}(C_2), \ldots, \mathtt{Ten}(C_k)$ are computed, a maximum-weight box of $P$ can be found in $O(k^2)$ time.*

**Proof.** An optimal box of $P$ which is not among $\mathrm{B}_{opt}(C_1), \ldots, \mathrm{B}_{opt}(C_k)$ can be computed as follows. Since the orthogonal projections of $\mathtt{Box}(C_1), \ldots, \mathtt{Box}(C_k)$ are pairwise disjoint in both axes, we can consider each cluster $C_i$ as a single point. Then we can run the algorithm corresponding to Theorem 2.2 making the following consideration. For any box set $X$ of $m$ clusters denoted from left to right $C'_1, C'_2, \ldots, C'_m$ the score $\mathtt{S}(X)$ must satisfy the following equations:

$$W(X) = \sum_{j=1}^{m} W(C'_j)$$

$$W\big(\mathrm{B}_L(X)\big) = \max_{j \in [1..m]} \sum_{i=1}^{j-1} W(C'_i) + W\big(\mathrm{B}_L(C'_j)\big)$$

$$W\big(\mathrm{B}_R(X)\big) = W\big(\mathrm{B}_R(C'_j)\big) + \max_{j \in [1..m]} \sum_{i=j+1}^{m} W(C'_i)$$

By using the operator $\oplus$, this can be guaranteed by considering for each cluster $C_i$ that $\mathrm{B}_L(C_i) = \mathrm{B}_{4,1}(C_i)$, and $\mathrm{B}_R(C_i) = \mathrm{B}_{2,3}(C_i)$. In the base case of the recursion there is at most one cluster above (resp. below) the strip $\mathcal{S}$ of the simplified Strip-Constrained Maximum-Weight Box problem for $P$. Consider the general setting in which there is a cluster $C'_1$ above $\mathcal{S}$ and a cluster $C'_2$ below $\mathcal{S}$, partitioning $\mathcal{S}$ into three boxes $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$. The other cases are similar and simpler to solve. Assume w.l.o.g. that $C'_1$ is located to the left of $C'_2$. Then the solution is the $\mathtt{Box}(\cdot)$ of one of the following ten sets, which represent all the forms

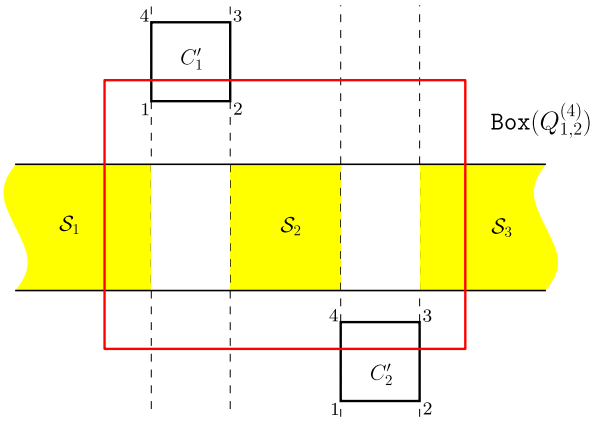


**Fig. 6.** The base case of the algorithm of Lemma 4.2.

of combining boxes of $\mathtt{Ten}(C'_1) \cup \mathtt{Ten}(C'_2)$ with boxes of $\mathtt{S}(P \cap \mathcal{S}_1)$, $\mathtt{S}(P \cap \mathcal{S}_2)$, and $\mathtt{S}(P \cap \mathcal{S}_3)$ (see Fig. 6):

$$
\begin{aligned}
Q_1^{(1)} &= \mathtt{B}_R(P \cap \mathcal{S}_1) \cup \mathtt{B}_1\big(C'_1\big)\\
Q_1^{(2)} &= \mathtt{B}_R(P \cap \mathcal{S}_1) \cup \mathtt{B}_{1,2}\big(C'_1\big) \cup \mathtt{B}_L\big(P \cap (\mathcal{S}_2 \cup \mathcal{S}_3)\big)\\
Q_1^{(3)} &= \mathtt{B}_2\big(C'_1\big) \cup \mathtt{B}_L\big(P \cap (\mathcal{S}_2 \cup \mathcal{S}_3)\big)\\
Q_2^{(1)} &= \mathtt{B}_R\big(P \cap (\mathcal{S}_1 \cup \mathcal{S}_2)\big) \cup \mathtt{B}_4\big(C'_2\big)\\
Q_2^{(2)} &= \mathtt{B}_R\big(P \cap (\mathcal{S}_1 \cup \mathcal{S}_2)\big) \cup \mathtt{B}_{3,4}\big(C'_2\big) \cup \mathtt{B}_L(P \cap \mathcal{S}_3)\\
Q_2^{(3)} &= \mathtt{B}_3\big(C'_2\big) \cup \mathtt{B}_L(P \cap \mathcal{S}_3)\\
Q_{1,2}^{(1)} &= \mathtt{B}_2\big(C'_1\big) \cup \mathcal{S}_2 \cup \mathtt{B}_4\big(C'_2\big)\\
Q_{1,2}^{(2)} &= \mathtt{B}_2\big(C'_1\big) \cup \mathcal{S}_2 \cup \mathtt{B}_{3,4}\big(C'_2\big) \cup \mathtt{B}_L(P \cap \mathcal{S}_3)\\
Q_{1,2}^{(3)} &= \mathtt{B}_R(P \cap \mathcal{S}_1) \cup \mathtt{B}_{1,2}\big(C'_1\big) \cup \mathcal{S}_2 \cup \mathtt{B}_4\big(C'_2\big)\\
Q_{1,2}^{(4)} &= \mathtt{B}_R(P \cap \mathcal{S}_1) \cup \mathtt{B}_{1,2}\big(C'_1\big) \cup \mathcal{S}_2\\
&\quad \cup \mathtt{B}_{3,4}\big(C'_2\big) \cup \mathtt{B}_L(P \cap \mathcal{S}_3)
\end{aligned}
$$

Since the algorithm runs over $k$ points the result holds from Theorem 2.2. □

Combining Theorem 3.2 with Lemma 4.2, joint with the fact that the algorithm of Theorem 3.2 can be generalized to compute $\mathtt{Ten}(P)$ in $\mathrm{SORT}(n) + O(\delta n)$ time, and also that

$$\sum_{i=1}^{k} n_i^2 + k^2 < \left(\sum_{i=1}^{k} n_i\right)^2 + n^2 = 2n^2,$$

we obtain the next result:

**Theorem 4.3.** *Given a cluster partition $(C_1, C_2, \ldots, C_k)$ of $P$, the* Maximum-Weight Box *problem admits a solution running in time within $O(\sum_{i=1}^{k} n_i\delta_i + k^2) \subset O(\sum_{i=1}^{k} n_i^2 + k^2) \subset O(n^2)$.*

Among all cluster partitions of $P$, only one is a diagonalization. Let $(C_1, C_2, \ldots, C_k)$ be a diagonalization of $P$. A *diagonalization tree* of $P$, denoted by D-tree, is a binary tree whose leaves are $C_1, C_2, \ldots, C_k$ from left to right and each internal node $u$ has two children $u_1$ and $u_2$ so that $(P(u_1), P(u_2))$ is a cluster partition of $P(u)$, where for each node $u$ set $P(u)$ denotes the union of the clusters in the leaves of the subtree rooted at $u$ (see Fig. 7).

**Lemma 4.4.** *A D-tree of $P$ requires $\Theta(n)$ space and can be built in $O(n \lg n)$ time.*

**Proof.** Let $p_1, p_2, \ldots, p_n$ denote the elements of $P$ sorted by $x$-coordinates, and let $p_{\pi_1}, p_{\pi_2}, \ldots, p_{\pi_n}$ denote the elements of $P$ sorted by $y$-coordinate. Considering the computation of permutation $\pi$ as a preprocessing, we now show that: If $P$ admits a cluster partition $(\{p_1, \ldots, p_s\}, \{p_{s+1}, \ldots, p_n\})$ then it can be determined in $O(\min\{s, n-s\})$ comparisons. Otherwise, if such a partition does not exist, then this can be decided in $O(n)$ time. For each index $i \in [1..n]$, let $M_L(i) = \max_{j\in[1..i]} \pi_j$, $m_L(i) = \min_{j\in[1..i]} \pi_j$, $M_R(i) = \max_{j\in[i..n]} \pi_j$, and $m_R(i) = \min_{j\in[i..n]} \pi_j$.

Observe that if $(\{p_1, \ldots, p_s\}, \{p_{s+1}, \ldots, p_n\})$ is a cluster partition of $P$, then index $s \in [1..n-1]$ satisfies $M_L(s) = s$ or $m_L(s) = n - s + 1$. Furthermore, $M_L(s) = s$ and $m_L(s) = n - s + 1$ are equivalent to $m_R(s+1) = s + 1$ and $M_R(s+1) = n - s$, respectively.

Then we can determine such a partition of $P$, if it exists, as follows: For $j = 1..\lfloor n/2 \rfloor$ decide if $(\{p_1, \ldots, p_j\}, \{p_{j+1}, \ldots, p_n\})$ is a cluster partition (i.e., $M_L(j) = j$ or $m_L(j) = n - j + 1$) or $(\{p_1, \ldots, p_{n-j}\}, \{p_{n-j+1}, \ldots, p_n\})$ is a cluster partition (i.e., $M_R(n - j + 1) = j$ or $m_R(n - j + 1) = n - j + 1$). Note that if $j > 1$ then $M_L(j)$, $m_L(j)$, $M_R(n - j + 1)$, and $m_R(n - j + 1)$ can all be computed in $O(1)$ time from $M_L(j-1)$, $m_L(j-1)$, $\pi_j$, $M_R(n - j + 2)$, $m_R(n - j + 2)$, and $\pi_{n-j+1}$. Therefore, if there is a cluster partition $(\{p_1, \ldots, p_s\}, \{p_{s+1}, \ldots, p_n\})$ of $P$ it is decided for $j = \min\{s, n - s\} \leqslant \lfloor n/2 \rfloor$, and thus determined in $O(j)$ time. If no such partition is found for any value of $j \in [1..\lfloor n/2 \rfloor]$, then the algorithm spends $O(n)$ time in total.

We can then build a D-tree of $P$ recursively as follows. Run the above algorithm for $P$. If a cluster partition $(\{p_1, \ldots, p_s\}, \{p_{s+1}, \ldots, p_n\})$ of $P$ exists, which was determined in $O(t)$ comparisons, where $t = \min\{s, n - s\}$, then create a root node and set as left child a D-tree of $\{p_1, \ldots, p_s\}$ and as right child a D-tree of $\{p_{s+1}, \ldots, p_n\}$. Otherwise, if $P$ does not admit such a partition, which was decided in $O(n)$ time, then create a leaf node with cluster $P$. This results in the next recurrence equation for the total number $T(n)$ of comparisons, where $1 \leqslant t \leqslant \lfloor n/2 \rfloor$:

$$T(n) = \begin{cases} O(t) + T(t) + T(n-t) & \\ \qquad n > 1, \text{ a cluster partition exists} & \\ O(n) \quad \text{otherwise.} & \end{cases}$$

W.l.o.g. assume that the constants in $O(t)$ and $O(n)$ in the recurrence are equal to one. Then we prove by induction that $T(n) \leqslant n + n \lg n$. The base case of the induction is the second line of the recurrence equation, where $n \leqslant n + n \lg n$ always holds. In the inductive case, we have $T(n) = t + T(t) + T(n-t) \leqslant n + t + t \lg t + (n-t)\lg(n-t) = n + n \lg n + n(t/n + H(t/n)) \leqslant n + n \lg n$, where $H(x) = x \lg(1/x) + (1-x)\lg(1/(1-x))$ is the binary entropy function, with $x = t/n$,

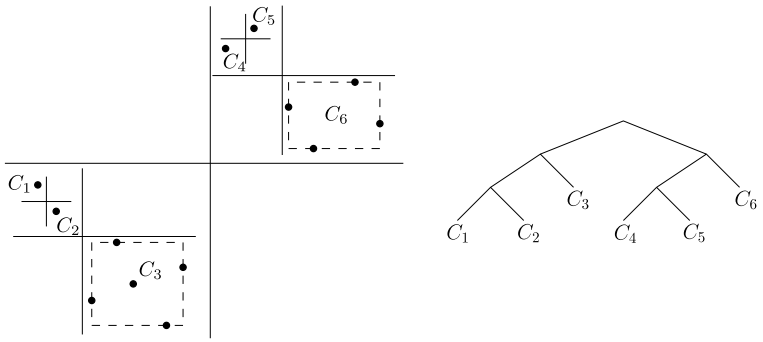


**Fig. 7.** A D-tree of the point set $P$ over the diagonalization $(C_1, \ldots, C_6)$.

and we use the analytic inequality $x \leqslant H(x)$, which holds at least for $x \leqslant 1/2$. Thus $T(n) \leqslant n + n \lg n$ and then $T(n)$ is within $O(n \lg n)$. One can see that this solution is tight by considering the case $t = n/2$. □

**Theorem 4.5.** *The* Maximum-Weight Box *problem admits a solution running in time within* $O(n \lg n + \sum_{i=1}^{k} n_i \delta_i + k) \subset O(n \lg n + \sum_{i=1}^{k} n_i^2) \subset O(n^2)$*, where $k$ is the size of the diagonalization $(C_1, \ldots, C_k)$ of $P$, and $n_i$ is the number of points of cluster $C_i$, which is composed of $\delta_i$ strips.*

**Proof.** Run the algorithm of Lemma 4.4 to build a D-tree of $P$ in $O(n \lg n)$ time. The $i$-th leaf of the D-tree from left to right contains cluster $C_i$. Compute $\texttt{Ten}(C_i)$ in time within $O(n_i \delta_i)$ using the algorithm from Theorem 3.2, assuming that $P$ is already sorted in $\text{SORT}(n) \subset O(n \lg n)$ time. Using a post-order traversal of the D-tree, for each internal node $u$ with child nodes $u_1$ and $u_2$, compute $\texttt{Ten}(P(u))$ in constant time from $\texttt{Ten}(P(u_1))$ from $\texttt{Ten}(P(u_2))$ by using Lemma 4.1. The result clearly follows from $\texttt{Ten}(u)$, where $u$ is the root of the D-tree. □

## 5. Upper and lower bounds in $d$ dimensions

In this section we study connections between the Maximum-Weight Box problem and others, deriving new upper and lower bounds for various related problems, in two and more dimensions.

*Connection to* Klee's Measure *problem and higher dimensions* Our $O(n^2)$ time algorithm for the Maximum-Weight problem is actually a special case of a more general result for a problem related to the well known Klee's Measure problem (computing the volume of a union of $n$ boxes).

In the $d$-dimensional Weighted Depth problem, we are given a set of $n$ weighted boxes in $\mathbb{R}^d$ and we want a point $p \in \mathbb{R}^d$ that maximizes the *depth*, defined as the sum of the weights of the boxes that contain $p$. All known algorithms for Klee's Measure problem can be modified to solve the Weighted Depth problem. In particular, Overmars and Yap's algorithm [15] runs in $O(n^{d/2} \lg n)$ time, Chan's algorithm [6] runs in $O(n^{d/2} 2^{O(\lg^* n)})$ time, and a new simple algorithm by Chan [7] runs in $O(n^{d/2})$ time.

The following result has not been noted before:

**Theorem 5.1.** *The* Maximum-Weight Box *problem in any constant dimension $d$ can be reduced to the* Weighted Depth *problem in dimension $2d$.*

**Proof.** Given a point set $P$ in $\mathbb{R}^d$, we map each point $p = (a_1, \ldots, a_d) \in P$ to a region $R_p$ in $\mathbb{R}^{2d}$, consisting of all $2d$-tuples $(x_1, \ldots, x_d, x'_1, \ldots, x'_d)$ such that $p$ lies inside the box with opposite corners $(x_1, \ldots, x_d)$ and $(x'_1, \ldots, x'_d)$; in other words, $R_p = \{(x_1, \ldots, x_d, x'_1, \ldots, x'_d) \mid [(x_1 \leqslant a_1 \leqslant x'_1) \vee (x'_1 \leqslant a_1 \leqslant x_1)] \wedge \cdots \wedge [(x_d \leqslant a_d \leqslant x'_d) \vee (x'_d \leqslant a_d \leqslant x_d)]\}$. We can decompose $R_p$ into a constant number of boxes in $\mathbb{R}^{2d}$, which will have weight $w(p)$. The maximum-weight box for $P$ corresponds to a point $(x_1, \ldots, x_d, x'_1, \ldots, x'_d)$ that has the maximum depth among these regions. □

According to the above theorem, our $O(n^2)$ result for the Maximum-Weight Box problem in two dimensions can also be deduced from Chan's latest result for the Weighted Depth problem in $d = 4$ dimensions [7]. In fact, the $O(n^2)$-time algorithm presented in this paper is inspired by Chan's algorithm [7], which is also based on a "divide–summarize-and-conquer" approach. The algorithm here is a more direct solution, avoiding the need to work explicitly in the 4-dimensional space, and also a pedagogical introduction to the algorithm running in time $\text{SORT}(n) + O(\delta n)$.

The above theorem also implies that the Maximum-Weight Box problem in $d$ dimensions can be solved in $O(n^d)$ time by Chan's new algorithm. Previously, only an $O(n^{2d-2} \lg n)$ time bound was reported [9].

*Polylogarithmic-factor speedups* Chan [7] also showed how to further speed up his algorithm by a polylogarithmic factor for the Weighted Depth problem, but only when the dimension is sufficiently large (in particular, not for $d = 4$).

However, in the unweighted case of the Depth problem, polylogarithmic speedup is possible [6,7] for any $d \geqslant 3$: the time can be reduced to $O((n^{d/2}/\lg^{d/2} n)(\lg \lg n)^{O(1)})$. This extends to the case where the weights are integers with absolute value bounded by $O(1)$, since we can replace a box with positive weight $c$ by $c$ copies of the box, and we can replace a box with negative weight $-c$ by $c$ copies of its complement (which can be decomposed into a constant number of boxes).

Therefore, we can solve the Maximum-Weight Box problem for the case of $+1$ and $-1$ weights in

$O((n^d/\lg^d n)(\lg\lg n)^{O(1)})$ time. The same bound thus follows for the Maximum Bichromatic Discrepancy problem. Previously, only an $O(n^2 \lg n)$ bound was known for $d = 2$ [9,10]. Similarly, by straightforward changes to incorporate $-\infty$ weights, the Maximum Box problem can be solved in $O((n^d/\lg^d n)(\lg\lg n)^{O(1)})$ time, improving the previous $O(n^d \lg^{d-2} n)$ time bound for $d \geqslant 3$ [1].

*Problem complexity* It is unknown whether $O(n^d)$ is the best possible time complexity for the Maximum-Weight Box problem, even in two dimensions: reducing the 3SUM problem to it, or proving an $\Omega(n^2)$ lower bound in some restricted model, would improve our understanding of a large family of problems in computational geometry. Note that if $d$ is part of the input then the Maximum-Weight Box problem is NP-hard, since it generalizes the Maximum Box problem [11]. In this regard, we can show the following:

**Theorem 5.2.** *The* Weighted Depth *problem in any constant dimension $d$ can be reduced to the* Maximum-Weight Box *problem in dimension $d$.*

**Proof.** We first reduce the Weighted Depth problem to a special case of the Weighted Depth problem where all the input boxes are "dominance" ranges of the form $(-\infty, b_1] \times \cdots \times (-\infty, b_d]$. To see this, for a given $i \in [1..d]$, we replace any input box $[a_1, b_1] \times \cdots \times [a_d, b_d]$ of weight $w$ with two boxes: $[a_1, b_1] \times \cdots \times [a_{i-1}, b_{i-1}] \times (-\infty, b_i] \times [a_{i+1}, b_{i+1}] \times \cdots \times [a_d, b_d]$ of weight $w$, and $[a_1, b_1] \times \cdots \times [a_{i-1}, b_{i-1}] \times (-\infty, a_i] \times [a_{i+1}, b_{i+1}] \times \cdots \times [a_d, b_d]$ of weight $-w$. By repeating this for each $i \in [1..d]$, each original box is replaced with $2^d$ boxes of the desired special form.

Now, given an instance of this special case of the Weighted Depth problem, we map each input box $b = (-\infty, b_1] \times \cdots \times (-\infty, b_d]$ to the point $p_b = (b_1, \ldots, b_d)$, of the same weight. We have the obvious property that $p_b$ lies inside the box $[x_1, \infty) \times \cdots \times [x_d, \infty)$ iff $(x_1, \ldots, x_d)$ lies inside $b$. We add an extra point at $(\infty, \ldots, \infty)$ with weight $M$ for a sufficiently large number $M$. The maximum-weight box containing the resulting point set must be of the form $[x_1, \infty) \times \cdots \times [x_d, \infty)$ because of this extra point, and so corresponds to a point of maximum depth of the given boxes. □

The above theorem implies the $W[1]$-hardness of the Maximum-Weight Box problem with respect to $d$, since Klee's Measure problem and the Weighted Depth problem are $W[1]$-hard [6]. It also implies the unlikeness of an algorithm that runs in time within $o(n^{d/2})$ (ignoring logarithmic factors) with current knowledge about Klee's Measure problem.

## References


[1] J. Backer, J. Keil, The mono- and bichromatic empty rectangle and square problems in all dimensions, in: Proc. 9th Latin American Theoretical Informatics Symposium (LATIN), 2010, pp. 14–25.

[2] J. Barbay, G. Navarro, P. Pérez-Lantero, Adaptive techniques to find optimal planar boxes, in: Proc. 24th Canadian Conference on Computational Geometry (CCCG), 2012, pp. 71–76.

[3] C. Bautista-Santiago, J.M. Díaz-Báñez, D. Lara, P. Pérez-Lantero, J. Urrutia, I. Ventura, Computing optimal islands, Oper. Res. Lett. 39 (4) (2011) 246–251.

[4] J. Bentley, Programming pearls: algorithm design techniques, Commun. ACM 27 (9) (1984) 865–873.

[5] S. Bereg, J.M. Díaz-Báñez, D. Lara, P. Pérez-Lantero, C. Seara, J. Urrutia, On the coarseness of bicolored point sets, Comput. Geom. 46 (1) (2013) 65–77.

[6] T.M. Chan, A (slightly) faster algorithm for Klee's measure problem, Comput. Geom. 43 (3) (2010) 243–250.

[7] T.M. Chan, Klee's measure problem made easy, in: Proc. 54th Annual Symposium on Foundations of Computer Science (FOCS), 2013, pp. 410–419.

[8] R. Cole, B. Mishra, J. Schmidt, A. Siegel, On the dynamic finger conjecture for splay trees. Part I: Splay sorting $\log n$-block sequences, SIAM J. Comput. 30 (1) (2000) 1–43.

[9] C. Cortés, J.M. Díaz-Báñez, P. Pérez-Lantero, C. Seara, J. Urrutia, I. Ventura, Bichromatic separability with two boxes: a general approach, J. Algorithms 64 (2–3) (2009) 79–88.

[10] D. Dobkin, D. Gunopulos, W. Maass, Computing the maximum bichromatic discrepancy, with applications to computer graphics and machine learning, J. Comput. Syst. Sci. 52 (3) (1996) 453–470.

[11] J. Eckstein, P. Hammer, Y. Liu, M. Nediak, B. Simeone, The maximum box problem and its application to data analysis, Comput. Optim. Appl. 23 (3) (2002) 285–298.

[12] X. Guo, Y. Yin, C. Dong, G. Yang, G. Zhou, On the class imbalance problem, in: Proc. 4th International Conference on Natural Computation, 2008, pp. 192–201.

[13] Y. Han, M. Thorup, Integer sorting in $O(n\sqrt{\log\log n})$ expected time and linear space, in: Proc. 33rd IEEE Symposium on Foundations of Computer Science (FOCS), 2012, pp. 135–144.

[14] Y. Liu, M. Nediak, Planar case of the maximum box and related problems, in: Proc. 15th Canadian Conference on Computational Geometry (CCCG), 2003, pp. 14–18.

[15] M.H. Overmars, C.-K. Yap, New upper bounds in Klee's measure problem, SIAM J. Comput. 20 (6) (1991) 1034–1045.

[16] S. Visa, A. Ralescu, Issues in mining imbalanced data sets — a review paper, in: Proc. 16th Midwest Artificial Intelligence and Cognitive Science Conference, 2005, pp. 67–73.