Fast Stabbing of Boxes in High Dimensions*

Franck Nielsen†

Abstract: We present in this paper a simple yet efficient algorithm for stabbing a set $\mathcal{S}$ of n axis-parallel boxes in d-dimensional space with c points in output-sensitive time $O(dn + n \log c)$ and linear space. Let c^* be the minimum number of points required to stab $\mathcal{S}$, then we prove that $c \leq \min\{\frac{c^* \overline{d}}{d!} + \frac{c^* \overline{d-1}}{(d-1)!} - 1, c^* \frac{(\log n+1)^{\overline{d-1}}}{(d-1)!}\}$, where $x^{\overline{m}}$ is the rising factorial power: $x^{\overline{m}} = \binom{x+m-1}{m}$. Since finding a minimal set of c^* points is NP-complete as soon as $d \geq 2$, we obtain a fast precision-sensitive heuristic for stabbing $\mathcal{S}$ in output-sensitive time and linear space. In the case of congruent or constrained isothetic boxes, our algorithm reports $c \leq 2^{d-1} c^*$ and $c = O(c^*)$ stabbing points, respectively. Moreover, we show that the bounds we get on c are tight and corroborate our results with some experiments. We also describe an optimal output-sensitive algorithm for finding an optimal stabbing point-set of intervals.

1 Setting the problem

Let $\mathcal{S}$ be a set of n d-dimensional geometric objects. We say that $\mathcal{S}$ is stabbed by k points if there exist k points so that each object of $\mathcal{S}$ contains at least one of these points. Given a set $\mathcal{S}$ as above, finding the minimum k so that $\mathcal{S}$ can be stabbed by k points has been shown to be NP-complete [FPT81] as soon as $d \geq 2$[1]. Therefore this problem is intractable even for small values of n. This problem is also referenced in the literature as the *covering set problem* (or dually as the *hitting set problem*) where it is transformed into an optimization problem by means of matrix formulations. Let $\mathcal{V} = \{S_i | i \in I\}$ be a collection of $v = |\mathcal{V}| = |I|$ subsets of $2^{\mathcal{S}}$ for a set $\mathcal{S}$ of n elements. We want to find a minimal covering collection, i.e. a subset $I' \subseteq I$ of indices so that $\mathcal{S} = \bigcup_{i \in I'} S_i$ with $|I'|$ as small as possible. More precisely, we want to minimize $e^T \times x = |I'|$ subject to $Ax \geq e$ for x a $\{0,1\}^v$-vector, $e = (1, ..., 1)$ and A a $(n \times v)$-binary matrix, each column of which is the incidence vector of one of the sets I_i, $1 \leq i \leq v$.

Some heuristics that give approximation of the minimum stabbing number c^* have been given. V. Chvátal [Chv79] gave a cubic greedy algorithm to find a cover set of size c such that $c \leq c^*(1+\log t)$ where t is the maximum column sum of A ($t \leq n$). D.S. Hochbaum [Hoc82] proposed another cubic algorithm with a cover set of size at most $c^* f$, where f is the maximum row sum of A. Interestingly, Bellare et al. [BGLR93] showed that no polynomial time algorithm can approximate the optimal solution within a factor of $(\frac{1}{8} - \epsilon) \log |\mathcal{S}|$, unless $NP \subseteq DTIME[n^{\log \log n}]$, where $\epsilon > 0$.

One major drawback from the computational geometry point of view is that these methods do not consider geometrical objects. (Although it has been shown that the intersection graph[2] of d-dimensional convex objects can be arbitrary as soon as $d \geq 3$ [Weg67].) This means that we have to supply matrix A. One way to proceed is to compute the whole arrangement of the objects and to consider all the incidence sets defined by vertices. More precisely, to each vertex we associate the set of objects containing it. Thus, the size of the matrix is $O(n^d) \times n$ and these algorithms require $O(n^{d+2})$ time and $O(n^{d+1})$ space.

D.S. Hochbaum and W. Maass [HM84] considered the case of geometric objects and give a polynomial-time approximation scheme (PTAS). Their method is innovative since it is general and takes into account the shape of the objects. Unfortunately, their algorithm is time-costly and considers sets of identical convex objects T, or dually covering sets of points with convex translates T^*. (T^* is the centrally symmetric convex object of T).

Many applications coming from VLSI design, image processing and point location have to deal with large inputs [TF80]. Recently, H. Brönninman and M.T. Goodrich [BG94] investigated these problems using the Vapnik-Chervonenkis dimension (VC-dimension). They obtain precision-sensitive set covers if the VC-dimension is bounded as it is generally the case when considering geometric objects. Their algorithm still relies on the fact that matrix A is computed beforehand.

In this paper, we are even more restrictive by considering the case of axis-parallel boxes in high dimensions (that are often considered in VLSI design, image processing and point location in d-dimensional Euclidean space); for example, we are given a set of points in E^d and some hypercube H_d. We want to associate to each point a hypercube that contains it so that we minimize the number of hypercubes. In other words, we want to

*Part of this work was done while the author was visiting at The University of Tokyo.

†INRIA, BP93, 06902 Sophia-Antipolis cedex (France) and University of Nice Sophia-Antipolis, Valrose (France). E-mail: Franck.Nielsen@sophia.inria.fr — http://www.inria.fr:/personnel/nielsen/frank.html

[1]More precisely, Fowler et al. [FPT81] showed that covering a set of points with fixed-size squares (the so-called BOX-COVER problem) is NP-complete as soon as $d > 1$.

[2]The intersection graph of a set of objects is defined as follows: we associate to each object a node and there exists an edge between two nodes iff the corresponding objects intersect.

cover the point set with a minimum number of patches, i.e. translates of H_d. Throughout the paper, the boxes are considered to be closed, i.e. points on the boundary of box B stab B. Our main algorithm, described in Section 3, will not require to compute the arrangement of the isothetic boxes[3]. We do not consider d as a constant in the sequel.

The paper is organized as follows:

In Section 2, we consider the case of a family of n intervals and give an optimal $\Theta(n(\log c^* + 1))$-time algorithm that gives an optimal stabbing set of c^* points. In Section 3, we describe our algorithm in higher dimensional spaces and study both its running time and its approximation factor. We refine the analysis for sets of congruent and constrained isothetic boxes. We corroborate our theoretical results with experiments. Finally, in Section 4, we give several guidelines for future research.

2 An optimal algorithm for stabbing intervals

2.1 Principle

Finding the minimum value c^* so that $\mathcal{S}$ can be stabbed with c^* points is easy and already known [HM84]. Consider the interval I that has the rightmost left endpoint q. I must be stabbed by a point and clearly, the best place to stab it is on its left endpoint q. We then remove all the intervals stabbed by q and loop until all the intervals are stabbed. We thus obtain an optimal set of c^* points that stab $\mathcal{S}$. A straightforward algorithm based on this fact has running time $O(nc^*)$ with linear space. We show below how an adequate preprocessing can yield an optimal output-sensitive algorithm in $\Theta(n(\log c^*+1))$ time and linear space.

2.2 Getting an output-sensitive algorithm

The methodology consists in grouping the intervals into groups and to preprocess each group in order to answer queries efficiently [Cha95, NY95]. Typically, our queries are of two kinds: "what are the intervals stabbed by a point q?" and "which interval has the rightmost left endpoint?". Moreover, we must be able to remove some of these intervals at some steps i, $1 \leq i \leq c^*$. We use the interval tree of McCreight [McC80, PS85] as the data-structure for answering these queries. Assume we know an estimate $\tilde{c^*}$ of c^*. Then, we group the n intervals into $\lceil \frac{n}{\tilde{c^*}} \rceil$ groups of size $\tilde{c^*}$ and preprocess each group into a static interval tree[4] for a total cost of $O(\lceil \frac{n}{\tilde{c^*}} \rceil \tilde{c^*} \log \tilde{c^*}) = O(n \log \tilde{c^*})$. At some step i, we find the rightmost left endpoint q_i of the remaining set of intervals and remove the n_i intervals stabbed by q_i from their corresponding groups. Thus the total cost of this step is $O(n_i \log \tilde{c^*} + \frac{n}{\tilde{c^*}} \log \tilde{c^*})$ (see [PS85], pp. 352–355). Therefore, the total cost of these c^* steps is $O(n \log \tilde{c^*} + \frac{n}{\tilde{c^*}} c^* \log \tilde{c^*})$ time since $\sum_{i=1}^{c^*} n_i = n$.

If we only want to test if $c^* > p$ we can derive a $O(n \log p)$-time algorithm by choosing $\tilde{c^*} = p$ and stopping the iterative process as soon as we have computed $i = \min\{c^*, p\}$ stabbing points. Note that our algorithm works in time $O(n \log c^*)$ if $c^* \leq \tilde{c^*} < c^{*2}$. Since we do not know c^* beforehand we iteratively estimate it by squaring our current estimate. We start with any arbitrary value for $\tilde{c^*}$, say $\tilde{c^*} = 2$, and square it until $\tilde{c^*} \geq c^*$. Thus, we obtain an $O(\sum_{i=0}^{\lceil \log \log c^* \rceil} n2^i) = O(n \log c^*)$ time algorithm. More details can be found in [Nie96].

Since verifying, if among n numbers, k are distinct requires $\Omega(n \log k)$ time on the real RAM [KS86], it follows that this lower bound also holds for the stabbing problem by reduction in linear time. Therefore, we obtain the following theorem:

Theorem 1 *Given a set $\mathcal{S}$ of n intervals, there exists an optimal output-sensitive algorithm that reports an optimal stabbing set of c^* points in optimal $\Theta(n \log c^*)$ time and linear space.*

3 The algorithm in higher dimensions

3.1 Principle

We describe below a "divide-and-conquer" strategy and show how we can get results on the approximation factor when dealing with axis-parallel boxes. Let $\mathcal{S}$ be a set of n d-dimensional boxes. Let $(O, \{x_1, ..., x_d\})$ denote the orthogonal frame of the d-dimensional Euclidean space. Each box can be viewed as the intersection of $2d$ half-spaces. A facet f of a box B is a $(d-1)$-dimensional box of the boundary ∂B supported by a hyperplane H_f. If H_f is defined by an equation of type $x_i = l$ for some real l then we say that f is a facet of *type* i. In other words, a facet of type i is perpendicular to the i-th axis. A box B can be viewed as the ordered cartesian product $\prod_{i=1}^{d} [r_i^-(B), r_i^+(B)]$ where $[r_i^-(B), r_i^+(B)]$ is the range of B along the i-th dimension. We say that box B is to the *left* (*right*) of $x_i = l$ if $r_i^+(B) < l$ (resp. $r_i^-(B) > l$). Let $X^{(i)}$ be the set of values defining the facets of type i, i.e. $X^{(i)} = \{x | \exists\ B \in \mathcal{S}$ such that $r_i^-(B) = x$ or $r_i^+(B) = x\}$ $(X^{(i)} = 2|\mathcal{S}|)$.

We describe below the algorithm (see also Figure 1):

Intervals (Basic case). If $\mathcal{S}$ is one-dimensional then apply the optimal algorithm of Section 2.2 for piercing this set of intervals.

[3] Computing the arrangement of a set of n isothetic boxes costs $O(n^d)$ time and space [PS85].

[4] In this context, static means that we know beforehand the $2\tilde{c^*}$ endpoints of each group. We only remove intervals and do not add new ones to that data-structure.

Partition. Let $x_n^{(d)}$ and $x_{n+1}^{(d)}$ be respectively the n-th and $(n+1)$-th greatest elements of $X^{(d)}$. Compute the 'median' $m = \frac{x_n^{(d)} + x_{n+1}^{(d)}}{2}$ of $X^{(d)}$ in linear time [BFP+72]. Partition $\mathcal{S}$ according to the hyperplane $H_m : (x_d = m)$ as follows:

- Let $\mathcal{S}_1$ be the set of boxes that do not cross H_m and are to the left of H_m.
- Let $\mathcal{S}_2$ be the set of boxes that do not cross H_m and are to the right of H_m.
- Let $\mathcal{S}_m$ be the set of boxes intersecting H_m.

Recursion. Stab the boxes of $\mathcal{S}_m$ by piercing the set of $(d-1)$-dimensional boxes: $\mathcal{S}'_m = \{B \cap H_m | B \in \mathcal{S}_m\}$.

Conquest. Stab recursively $\mathcal{S}_1$ and $\mathcal{S}_2$.

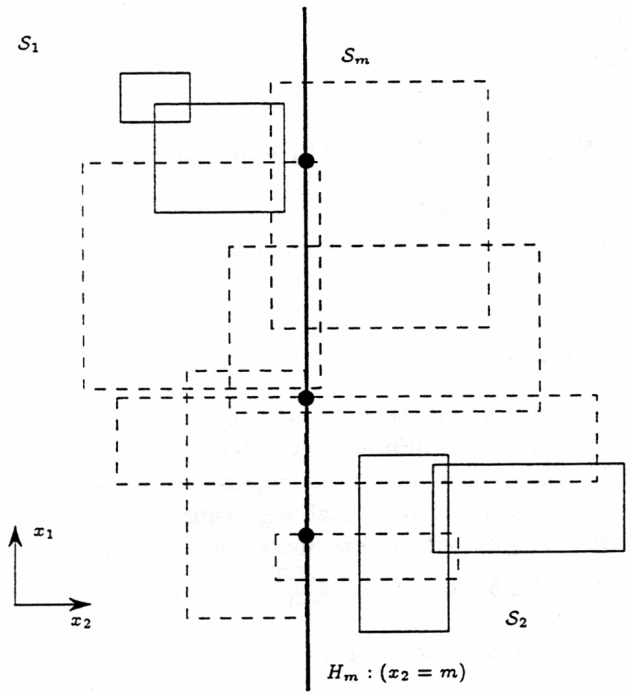

Figure 1: Partition of $\mathcal{S}$ into three subsets depending on their location with respect to the hyperplane $H_m : (x_d = m)$. We denote by $\mathcal{S}'_m$ the set of $(d-1)$-dimensional boxes $\mathcal{S}_m \cap H_m$.

Let $t(\mathcal{S})$ and $c(\mathcal{S})$ be respectively the running time of the algorithm and the number of stabbing points delivered by this algorithm. Sometimes, when we want to specify the dimension d of $\mathcal{S}$, we put in subscript of these notations a d. Thus, $t_k(\mathcal{S})$ and $c_k(\mathcal{S})$ denote respectively the running time and the output-size of our algorithm for a set of k-dimensional isothetic boxes S. Denote by $c^*(\mathcal{S})$ the minimum number of stabbing points of set $\mathcal{S}$ of d-dimensional boxes. In the sequel, d is not assumed to be a constant.

Our algorithm relies on the following simple facts:

Monotonicity. For any object O, $c^*(\{O\} \cup \mathcal{S}) \geq c^*(\mathcal{S})$.

Additive rule. Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be two subsets of objects so that $\forall I_1 \in \mathcal{I}_1$, $\forall I_2 \in \mathcal{I}_2, I_1 \cap I_2 = \emptyset$ then $c^*(\mathcal{I}_1 \cup \mathcal{I}_2) = c^*(\mathcal{I}_1) + c^*(\mathcal{I}_2)$.

Cutting rule. Let $\mathcal{S}$ be a set of boxes and H a hyperplane of type i, with $1 \leq i \leq d$. Then, $c^*(\mathcal{S}_H) = c^*(\mathcal{S}'_H)$ where $\mathcal{S}'_H = \{B \cap H | B \in \mathcal{S}\}$ and $\mathcal{S}_H = \{B | B \cap H \neq \emptyset\}$.

We study below the number $c_d(\mathcal{S})$ of points returned by our algorithm:

$$c_d(\mathcal{S}) = \begin{cases} c_{d-1}(\mathcal{S}_m \cap H_m) \text{ if } \mathcal{S}_1 = \mathcal{S}_2 = \emptyset, \\ c_{d-1}(\mathcal{S}_m \cap H_m) + c_d(\mathcal{S}_1) + c_d(\mathcal{S}_2) \quad \text{otherwise.} \end{cases}$$

Let us prove by induction on the lexicographically ordered vector (d, n) that $c_d(\mathcal{S}) \leq \frac{c^*(\mathcal{S})^{\overline{d}}}{d!} + \frac{c^*(\mathcal{S})^{\overline{d-1}}}{(d-1)!} - 1$, where $x^{\overline{m}} = \binom{x+m-1}{m}$.

Proof.

For $d = 1$, Section 2.2 describes an optimal algorithm so that $c_1(\mathcal{S}) = c^*(\mathcal{S}) \leq c^*(\mathcal{S}) + 1 - 1$ since $x^{\overline{0}} = 1$ by convention (finite calculus rules may be found in [GKP94]). If $|\mathcal{S}| = n = 1$ then $c_d(\mathcal{S}) = c^*(\mathcal{S}) = 1 \leq \frac{1^{\overline{d}}}{d!} + \frac{1^{\overline{d-1}}}{(d-1)!} - 1$.

Otherwise ($d > 1$ and $n > 1$), there are two cases depending on whether $\mathcal{S}_1, \mathcal{S}_2 = \emptyset$ or not (observe that $|\mathcal{S}_1| = |\mathcal{S}_2|$).

If $\mathcal{S}_1 = \mathcal{S}_2 = \emptyset$ then we have:

$$c(\mathcal{S}) \leq \frac{c^*(\mathcal{S})^{\overline{d-1}}}{(d-1)!} + \frac{c^*(\mathcal{S})^{\overline{d-2}}}{(d-2)!} - 1 \leq \frac{c^*(\mathcal{S})^{\overline{d}}}{d!} + \frac{c^*(\mathcal{S})^{\overline{d-1}}}{(d-1)!} - 1,$$

since $\frac{(c^*(\mathcal{S})+d-1)(c^*(\mathcal{S})+d-2)}{d(d-1)} \geq 1$ (recall that $c^*(\mathcal{S}) \geq 1$).

If $\mathcal{S}_1, \mathcal{S}_2 \neq \emptyset$ then we have:

$$c_d(\mathcal{S}) = c_d(\mathcal{S}_1) + c_d(\mathcal{S}_2) + c_{d-1}(\mathcal{S}'_m),$$

$$c_d(\mathcal{S}) \leq \frac{c^*(\mathcal{S}_1)^{\overline{d}} + c^*(\mathcal{S}_2)^{\overline{d}}}{d!} + \frac{c^*(\mathcal{S}_1)^{\overline{d-1}} + c^*(\mathcal{S}_2)^{\overline{d-1}}}{(d-1)!} + \left(\frac{c^*(\mathcal{S})^{\overline{d-1}}}{(d-1)!} + \frac{c^*(\mathcal{S})^{\overline{d-2}}}{(d-2)!}\right) - 3,$$

with $1 \leq c^*(\mathcal{S}_1), c^*(\mathcal{S}_2)$ and $c^*(\mathcal{S}_1) + c^*(\mathcal{S}_2) \leq c^*(\mathcal{S})$, since $c^*(\mathcal{S}'_m) = c^*(\mathcal{S}_m) \leq c^*(\mathcal{S})$. Since the rising factorial power is a convex function, the right hand side of the last inequality is maximized[5] for $c^*(\mathcal{S}_1) = 1$ and

[5] Let $f(\cdot)$ be a convex function defined on range $[a, b]$ then $\max_{a \leq c \leq b}\{f(c)\} \leq \max\{f(a), f(b)\}$. Let $g_m(x) = x^{\overline{m}}$. $g_m(x)$ is convex on $[0, +\infty)$ for $m \geq 0$. Define $f(x) = g_m(x) + g_m(c - x) + g_{m-1}(c)$ for $1 \leq x \leq c - 1$. $f(x)$ is convex on $[1, c-1]$ and therefore $\max_{1 \leq x \leq c-1}\{f(x)\} \leq f(1)$.

$c^*(\mathcal{S}_2) = c^*(\mathcal{S}) - 1$ (or the other way around). It follows that:

$$c_d(\mathcal{S}) \leq \frac{(c^*(\mathcal{S})-1)^{\overline{d}}}{d!} + \frac{(c^*(\mathcal{S})-1)^{\overline{d-1}}}{(d-1)!}$$
$$+\left(\frac{c^*(\mathcal{S})^{\overline{d-1}}}{(d-1)!} + \frac{c^*(\mathcal{S})^{\overline{d-2}}}{(d-2)!}\right) - 3 + \frac{1^{\overline{d}}}{d!} + \frac{1^{\overline{d-1}}}{(d-1)!},$$

$$c_d(\mathcal{S}) \leq \frac{c^*(\mathcal{S})^{\overline{d-1}}}{d!}(c^*(\mathcal{S}) - 1 + d) +$$
$$\frac{c^*(\mathcal{S})^{\overline{d-2}}}{(d-1)!}(c^*(\mathcal{S}) - 1 + d - 1) - 1,$$

and therefore

$$c_d(\mathcal{S}) \leq \frac{c^*(\mathcal{S})^{\overline{d}}}{d!} + \frac{c^*(\mathcal{S})^{\overline{d-1}}}{(d-1)!} - 1.$$

□

Remark. For small values of $c^*(\mathcal{S})$ we get an approximation factor that is polynomial in d. As an example, consider $c^*(\mathcal{S}) = 3$ then we have $c(\mathcal{S}) \leq d(d+2)$. (The approximation scheme of Hochbaum and Maass [HM84] has an exponential dependence in d). However, when $c^*(\mathcal{S})$ is large (say $c^*(\mathcal{S}) > dn^{\frac{1}{d}}$) we get the trivial bound $c_d(\mathcal{S}) \leq n$. It does not reflect the dichotomy process. Therefore, by studying the relationships between $c(\mathcal{S})$ and $|\mathcal{S}| = n$, one might prove that $c(\mathcal{S}) \leq c^*(\mathcal{S})\frac{(\log n+1)^{\overline{d-1}}}{(d-1)!}$ (see [Nie96] for full details).

Let us now analyze the time spent by this algorithm for reporting the $c_d(\mathcal{S})$ stabbing points.

We have:

$$t_d(\mathcal{S}) = \begin{cases} 0 & \text{if } \mathcal{S} = \emptyset, \\ An + t_{d-1}(\mathcal{S}'_m) + t_d(\mathcal{S}_1) + t_d(\mathcal{S}_2) & \text{otherwise.} \end{cases}$$

with A some constant related to the implementation of the partition scheme [BFP+72].

We prove below by induction on the lexicographically ordered vector (d, n) that $t_d(\mathcal{S}) \leq A'(n \log c(\mathcal{S}) + dn)$ $(\star)$.

Proof. If $d = 1$ then we proved in Section 2.2 an $O(n \log c(\mathcal{S}) + n)$-time algorithm. Therefore, $t_1(S) \leq Bn(\log c(\mathcal{S}) + 1) \leq A'n(\log c(\mathcal{S}) + 1)$ for $A' \geq B$. If $n = 1$ then $t_d(\mathcal{S}) \leq Ad \leq A'd$ and $(\star)$ holds trivially for $A' \geq A$.

Otherwise ($d > 1$ and $n > 1$), consider the two cases depending on whether $\mathcal{S}_1, \mathcal{S}_2 = \emptyset$ or not: In the former case, we have $t_d(\mathcal{S}) = t_{d-1}(\mathcal{S}) + An$. Thus, we get

$$t_d(\mathcal{S}) \leq A'n\Big((d-1)+\log c(\mathcal{S})\Big)+An \leq A'n(d+\log c(\mathcal{S})).$$

In the latter case ($\mathcal{S}_1, \mathcal{S}_2 \neq \emptyset$) , we have:

$$t_d(\mathcal{S}) = An + t_d(\mathcal{S}_1) + t_d(\mathcal{S}_2) + t_{d-1}(\mathcal{S}'_m),$$

with $1 \leq |\mathcal{S}_1|, |\mathcal{S}_2| \leq \frac{n}{2}$, $|\mathcal{S}_1| + |\mathcal{S}_2| + |\mathcal{S}'_m| = n$ and $c(\mathcal{S}) = c(\mathcal{S}_1) + c(\mathcal{S}_2) + c(\mathcal{S}'_m)$. Let $n' = |\mathcal{S}_1| = |\mathcal{S}_2|$ ($|\mathcal{S}'_m| = n - 2n'$). Hence,

$$t_d(\mathcal{S}) \leq A'\Big(n + n' \log c(\mathcal{S}_1) + n'd + n' \log c(\mathcal{S}_2) + n'd +$$
$$(n-2n') \log c(\mathcal{S}'_m) + (n-2n')(d-1)\Big),$$

$$t_d(\mathcal{S}) \leq A'\Big(nd + 2n' + n'(\log c(\mathcal{S}_1) + \log c(\mathcal{S}_2)) +$$
$$(n-2n') \log c(\mathcal{S}'_m)\Big).$$

But $\log c(\mathcal{S}_1) + \log c(\mathcal{S}_2)$ is maximized when $c(\mathcal{S}_1) = c(\mathcal{S}_2) \leq \frac{c(\mathcal{S})}{2}$.

Therefore, we get

$$t_d(\mathcal{S}) \leq A'\Big(dn + 2n' \log c(\mathcal{S}) + (n-2n') \log c(\mathcal{S}'_m)\Big),$$

with $c(\mathcal{S}'_m) \leq c(\mathcal{S})$.

Finally, we get $t_d(\mathcal{S}) \leq A'n(d + \log c(\mathcal{S}))$.

□

We have shown:

Theorem 2 *Let $\mathcal{S}$ be a set of n d-dimensional boxes. Denote by c^* the minimum number of stabbing points of $\mathcal{S}$. Then, there exists an output-sensitive algorithm that reports a set of c stabbing points of $\mathcal{S}$ in time $O(dn + n \log c)$ with linear space whose approximation c is bounded by $\min\{\frac{c^{*\overline{d}}}{d!} + \frac{c^{*\overline{d-1}}}{(d-1)!} - 1, c^*\frac{(\log n+1)^{\overline{d-1}}}{(d-1)!}\}$.*

Note that $\mathcal{S}$ is of size $O(dn)$.

3.2 Congruent or constrained boxes

A box $B = [b, t]$ is defined by its bottommost corner $b = (b_1, ..., b_d)$ and its topmost corner $t = (u_1, ..., u_d)$. Let $\mathcal{S}$ be a collection of n boxes:

$$\mathcal{S} = \{[(b_{i,1}, ..., b_{i,d}), (u_{i,1}, ..., u_{i,d})] | 1 \leq i \leq n\}.$$

Let $B_1 = \max_{i,j=1..n}\{\frac{u_{i,1}-b_{i,1}}{u_{j,1}-b_{j,1}}\}$, ..., $B_d = \max_{i,j=1..n}\{\frac{u_{i,d}-b_{i,d}}{u_{j,d}-b_{j,d}}\}$, $1 \leq i \leq d$. We say that $\mathcal{S}$ is *constrained* if $B_i = O(1)$ for $1 \leq i \leq d$. In [Nie96], we exhibit an example where our algorithm reaches its worst-case performance. However, in order to build it, we did consider stretched boxes, i.e. non-constrained boxes.

Lemma 3 *Let $\mathcal{S}$ be a set of n congruent isothetic d-dimensional boxes. Then, our algorithm guarantees that $c(\mathcal{S}) \leq 2^{d-1}c^*(\mathcal{S})$.*

Proof. W.l.o.g. consider the case of (unit) hypercubes. $\mathcal{S}$ is a collection of n congruent hypercubes. We prove by induction below that $c(\mathcal{S}) \leq 2^{d-1}c^*(\mathcal{S})$. Section 2.2 shows that the algorithm ensures $c_1(\mathcal{S}) = c^*(\mathcal{S})$ for (unit) intervals.

Otherwise, let $\mathcal{S}$ be a set of n d-boxes. Consider the ordered sequence (left to right) of cutting hyperplanes of type d: $(H_m(1) : x_d = a_1), ..., (H_m(k) : x_d = a_k)$ with the associated partition of the hypercubes $\mathcal{S}'_m(1), ..., \mathcal{S}'_m(k)$ (see the *partition* step of the algorithm). Clearly, we have $a_{i+1} - a_i > \frac{1}{2}, 1 \leq i \leq k-1$ for the case of unit hypercubes. Therefore $\mathcal{S}'_m(i) \cap \mathcal{S}'_m(j) = \emptyset$ if $|i-j| \geq 2$. We have:

$$c_d(\mathcal{S}) = \sum_{i=1}^{k} c_{d-1}(\mathcal{S}'_m(i)) \leq 2^{d-2} \sum_{i=1}^{k} c^*(\mathcal{S}'_m(i)).$$

We can decompose the last sum taking into account the parity of i as follows:

$$c_d(\mathcal{S}) \leq 2^{d-2}\Big(\sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} c^*(\mathcal{S}'_m(2i)) + \sum_{i=0}^{\lceil \frac{k}{2} \rceil - 1} c^*(\mathcal{S}'_m(2i+1))\Big).$$

But $\sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} c^*(\mathcal{S}'_m(2i)) = c^*(\cup_{i=1..\lfloor \frac{k}{2} \rfloor} \mathcal{S}'_m(2i)) \leq c^*(\mathcal{S})$ and $\sum_{i=0}^{\lceil \frac{k}{2} \rceil - 1} c^*(\mathcal{S}'_m(2i+1)) = c^*(\cup_{i=0..\lceil \frac{k}{2} \rceil - 1} \mathcal{S}'_m(2i+1)) \leq c^*(\mathcal{S})$ since both $S'_m(2i) \cap S'_m(2j) = \emptyset$ and $S'_m(2i+1) \cap S'_m(2j+1) = \emptyset$ as soon as $i \neq j$.

Therefore, we get:

$$c_d(\mathcal{S}) \leq 2^{d-2} \times 2c^*(\mathcal{S}) \leq 2^{d-1}c^*(\mathcal{S}).$$

□

In [HM84], Hochbaum and Maass also considered this problem (in its dual form however) and gave an $O(l^d n^{2l^d+1})$-time algorithm (a polynomial time approximation scheme) which ensures that $c(\mathcal{S}) \leq (1+\frac{1}{l})^d c^*(\mathcal{S})$ for a given integer $l \geq 1$. Thus, for $l = 1$ it yields an $O(n^3)$-time algorithm with performance ratio 2^d.

Using the same technique as above, we get the following lemma:

Lemma 4 *Let $\mathcal{S}$ be a collection of n d-dimensional constrained boxes with $B_1, ..., B_d$ defined as above. Then, our algorithm will return $c(\mathcal{S})$ stabbing points so that $c(\mathcal{S}) \leq (\prod_{i=2}^{d} \lceil 2B_i \rceil) c^*(\mathcal{S}) = O(c^*(\mathcal{S}))$.*

We may assume w.l.o.g. that $B_1 = \max_{i=1..d}\{B_i\}$. Otherwise, we make a simple rotation of the orthogonal frame in linear time. This also means that we may have a direction where the projected boxes are not constrained since we are able to solve exactly the problem in one dimension (see Section 2.2).

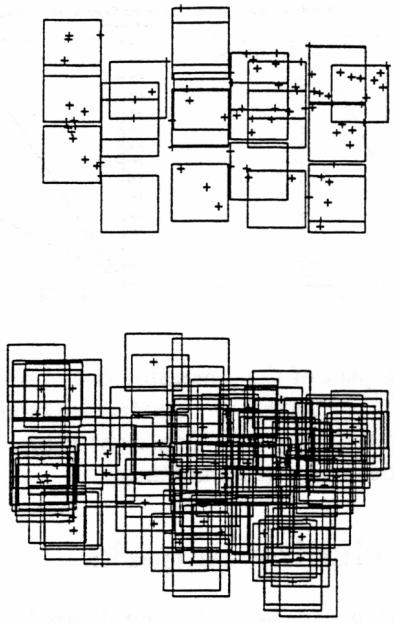

Figure 2: Finding a covering of 120 cities of the U.S.A. Our algorithm reports a set of 25 unit squares covering this 120 cities.

3.3 Experimental results

We did the implementation in C++ using the LEDA[6] and CGAL[7] libraries. The code length is about 1000 lines. As an application, we considered the following problem (already mentioned in the introduction): given a set of n cities $\mathcal{C} = \{C_1 = (x_1, y_1), ..., C_n = (x_n, y_n)\}$, we wish to cover them by a minimum number of copies of a 'unit' square $S = [-r, r] \times [-r, r]$ of side length $2r$. We associate to each city $C_i = (x_i, y_i)$ the square $S_i = [x_i - r, x_i + r] \times [y_i - r, y_i + r]$ for $1 \leq i \leq n$. Let $\mathcal{S} = \{S_1, ..., S_n\}$. Now, we use the fact that $\mathcal{C}$ can be covered with k translated copies of S if and only if $\mathcal{S}$ can be stabbed with k points. We ran our program on 120 cities of the United States of America (the input file is freely distributed and may be found in the *Stanford GraphBase* [Knu93]). The results are depicted in Figure 2.

Figure 3 shows some experiments for sets $\mathcal{S}$ (with $|\mathcal{S}| = 10000$) that are 20-pierceable, i.e. $c^*(\mathcal{S}) = 20$. The left chart shows the number of stabbing points reported by our algorithm in case of congruent/nonconstrained boxes. The right chart depicts the running time of our algorithm. (We took the average over 20 trials). More experiments concerning the number of different configurations are described in [Nie96].

[6] Library for Efficient Data-structures and Algorithms. Max-Planck Institut für Informatik, Im Stadtwald – 66123 Saarbrücken – Germany.

[7] see "The CGAL Kernel User Manual" – INRIA Sophia-Antipolis (France).

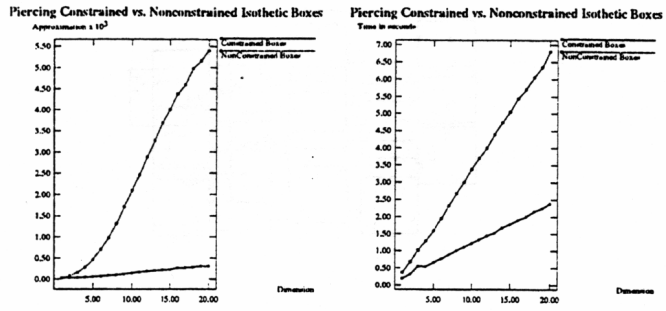

Figure 3: Impact of the dimension over a set $\mathcal{S}$ ($c^*(\mathcal{S}) = 20$ and $|\mathcal{S}| = 10000$) of constrained/nonconstrained d-boxes for $1 \leq d \leq 20$. The right chart exhibits the running time of our implementation

4 Concluding remarks

Our algorithm can be easily parallelized onto PRAM computers in order to gain efficiency (see [AL93]). This paper raises some open problems:

- the hardness of approximation of 'constrained' boxes compared with 'nonconstrained' boxes inside the polynomial-time hierarchy of problems.
- finding a PTAS (polynomial-time approximation scheme) for the case of 'nonconstrained' isothetic boxes. (Hochbaum and Maass PTAS [HM84] only works for congruent boxes).

Another aspect of this problem that is currently being investigated is to give efficient algorithms to detect whether a set of objects is k-pierceable or not for small values of k and several classes of convex objects [KN96].

Acknowledgements. I would like to thank Hervé Brönnimann, Jean-Daniel Boissonnat, Monique Teillaud and Mariette Yvinec for helpful discussions and comments.

References

[AL93] S. G. Akl and K. A. Lyons. *Parallel Computational Geometry.* Prentice-Hall, 1993.

[BFP+72] M. Blum, R.W. Floyd, V.R. Pratt, R.L. Rivest, and Tarjan. Time bounds for selection. *J. Comput. Syst. Sci.*, 7, 1972.

[BG94] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite VC-dimension. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 293–302, 1994.

[BGLR93] M. Bellare, S Goldwasser, C. Lund, and A. Russel. Efficient probabilistically checkable proofs and applications to approximation. In *Proc. 25th Annu. Symp. Theory Comput.*, pages 294–304, 1993.

[Cha95] M. Y. Chan. Output-Sensitive Results on Convex Hulls, Extreme Points, and Related Problems. In *Proc. 11th ACM Symp. on Comput. Geometry*, 1995.

[Chv79] V. Chvátal. A greedy heuristic for the set-covering problem. *Math. Oper. Res.*, 4:233–235, 1979.

[FPT81] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Inform. Process. Lett.*, 12(3):133–137, 1981.

[GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics.* Addison-Wesley Publishing Company, New York, 1994.

[HM84] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 31:130–136, 1984.

[Hoc82] D. S. Hochbaum. Approximation Algorithms of the Set Covering and Vertex Cover Problems. *SIAM J. Comput.*, 11(3):555 – 556, August 1982.

[KN96] M. J. Katz and F. Nielsen. On piercing Sets of Objects. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, to appear.

[Knu93] D. E. Knuth. *The Stanford Graphbase*, volume 1 of *A Platform for Combinatorial Computing.* Addison-Wesley, Reading, MA, 1993.

[KS86] D. G. Kirkpatrick and R. Seidel. The ultimate planar convex hull algorithm? *SIAM J. Comput.*, 15:287–299, 1986.

[McC80] E. M. McCreight. Efficient algorithms for enumerating intersecting intervals and rectangles. Report CSL-80-9, Xerox Palo Alto Res. Center, Palo Alto, CA, 1980.

[Nie96] F. Nielsen Fast Stabbing of Boxes in High Dimension. *INRIA report 2854.* INRIA Sophia-Antipolis, France, 1996.

[NY95] F. Nielsen and M. Yvinec. Output-Sensitive Convex Hull Algorithms of Planar Convex Objects. *INRIA report 2575.* INRIA Sophia-Antipolis, France, 1995.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.

[TF80] S.L. Tanimoto and R.J. Fowler. Covering image subsets with patches. In *Proc. of the 5th international conference on pattern recognition*, pages 835–839, 1980.

[Weg67] G. Wegner. *Eigenschaften der Nerven homologish-einfacher Familien im $\mathcal{R}^n$.* PhD thesis, Göttingen, 1967.