Approximating the piercing number for unit-height rectangles

Timothy M. Chan* Abdullah-Al Mahmood†

Abstract

The piercing problem seeks the minimum number of points for a set of objects such that each object contains at least one of the points. We present a polynomial-time approximation scheme (PTAS) for the piercing problem for a set of axis-parallel unit-height rectangles. We also examine the problem in a dynamic setting and show how to maintain a factor-2 approximation under insertions in logarithmic amortized time, by solving an incremental version of the maximum independent set problem for interval graphs.

1 Introduction

Given n objects in $\mathbb{R}^d$, a piercing set is a set of points such that each object contains at least one of the points in the set. The minimum cardinality of a piercing set is known as the piercing number. The problem of computing the piercing number is analogous to the hitting set problem [7] and is NP-hard even for the special case where the objects are axis-aligned unit squares in $\mathbb{R}^2$ [6]. (The piercing problem in this case is equivalent to covering a given set of points with the minimum number of unit squares.) Focus is thus turned towards approximation algorithms. The general hitting-set problem allows for a logarithmic approximation factor [3]; in this paper, we are interested in obtaining better approximation results in geometric settings.

Hochbaum and Maass [9] gave a factor-$(1+\epsilon)$ algorithm (a PTAS) for axis-aligned unit squares in $\mathbb{R}^2$ running in $n^{O(1/\epsilon^2)}$ time. The time bound was later improved to $n^{O(1/\epsilon)}$ by Feder and Greene [5] and Gonzalez [8]. We are interested in identifying larger classes of objects that admit similar PTAS results. For instance, Chan [2] recently provided a PTAS that runs in $n^{O(1/\epsilon^2)}$ time for squares of possibly different sizes (and more generally any collection of "fat" objects). See also [4, 13] for earlier results on piercing. In this paper we consider the case of axis-parallel unit-height rectangles and present a new PTAS that solves the problem in $n^{O(1/\epsilon^2)}$ time.

*School of Computer Science, University of Waterloo, tmchan@uwaterloo.ca. Supported by an NSERC grant and a Premier's Research Excellence Award.

†School of Computer Science, University of Waterloo, amahmood@uwaterloo.ca.

We also investigate the problem in a dynamic setting where insertions of new rectangles are allowed. Katz *et al.* considered the dynamic problem for 1-dimensional intervals and provided an exact algorithm that can support both insertions and deletions in $O(p \log n)$ where p denotes the piercing number [10]. Katz *et al.* also gave a $(1+\epsilon)$-approximation algorithm with a better update time bound of $O((1/\epsilon)\log n)$. These results imply a factor-2 and factor-$(2+\epsilon)$ approximation algorithm for 2-dimensional unit-height rectangles with $O(p \log n)$ and $O((1/\epsilon)\log n)$ update times respectively. We show that in the insertion-only setting, the amortized update time for the 1-dimensional exact algorithm and the 2-dimensional factor-2 algorithm can be reduced from $O(p \log n)$ to $O(\log n)$. Our dynamic 1-dimensional result might be of independent interest, as the piercing problem for intervals is equivalent to finding maximum independent sets in interval graphs (an often-studied "activity selection" problem [3]). See also [11] for another logarithmic-time result for the 1-dimensional problem for a special kind of "endpoint exchange" updates.

2 A Factor-$(1+\epsilon)$ Algorithm

Lemma 1 *For a set of n unit-height axis-parallel rectangles, sorted according to their right (or left) boundaries, a factor-2 approximation of its piercing number can be computed in $O(n)$ time.*

Proof. For each horizontal grid line $y = i$, the piercing number of the intervals formed by rectangles intersecting that line can be computed in linear time, by a standard greedy algorithm that examines the right endpoints from left to right (see the next section for details). The approximation is obtained by adding all the piercing numbers computed. Since each rectangle can be intersected by at most two of the lines, the overall running time is linear.

Let $P^{(i)}$ be the set of piercing points computed for the line $y = i$. Let Z be a minimum set of piercing points for the entire set of rectangles. Let $Z^{(i)}$ be the subset of all points in Z inside the strip $i-1 \le y < i$. Since rectangles intersecting $y = i$ are contained in the strip $i-1 \le y < i+1$, $|P^{(i)}| \le |Z^{(i)}| + |Z^{(i+1)}|$. Therefore, $\sum_i |P^{(i)}| \le 2|Z|$, as desired. □

Lemma 2 *For constant integers $k, k' \ge 1$, if a set of n unit-height axis-parallel rectangles can be stabbed by*

k horizontal lines, a factor-$(1+\frac{1}{k'})$ approximation of the piercing number can be computed in $O(n^{4kk'+2k-1})$ time.

Proof. Let R be the given set of rectangles. Let $a_1 \ldots a_{m-1}$ be the x-coordinates of the corners of the rectangles, sorted in increasing order, with $a_m = \infty$. The following algorithm computes a piercing set P of R:

1. Set $P \leftarrow \emptyset$ and $\ell \leftarrow -\infty$.
2. For $i \leftarrow 1$ to m do
    3. Let R_i be the subset of all rectangles intersected by the vertical line $x = a_i$.
    4. Also let R'_i be the subset of all rectangles lying entirely inside the strip $\ell \le x < a_i$. Denote this strip by σ_i.
    5. Compute a lower bound on the piercing number of R'_i using Lemma 1. If the lower bound is at least kk', or $i = m$, then
        6. compute the exact minimum piercing set P_i of R_i by a 1-dimensional algorithm;
        7. compute the exact minimum piercing set P'_i of R'_i by exhaustive search;
        8. set $P \leftarrow P \cup P_i \cup P'_i$ and $\ell \leftarrow a_i$.

Let $\mathcal{I}$ be the set of values of i for which steps 6–8 are executed. The algorithm in effect partitions R into subsets R_i and R'_i ($i \in \mathcal{I}$). Let Z be a minimum piercing set of R. By assumption, $|P_i| \le k$ for every i, with $|P_m| = 0$. Since the rectangles in R'_i are contained in σ_i, $|P'_i| \le |Z \cap \sigma_i|$, implying that $\sum_{i\in\mathcal{I}} |P'_i| \le |Z|$. Since $|P'_i| \ge kk'$ for all $i \in \mathcal{I}$ except for $i = m$, we have $|\mathcal{I}| - 1 \le \frac{|Z|}{kk'}$. Thus,

$$
\begin{aligned}
|P| &= \sum_{i\in\mathcal{I}} |P_i| + \sum_{i\in\mathcal{I}} |P'_i| \\
&\le k(|\mathcal{I}|-1) + |Z| \;\le\; (1+1/k')|Z|,
\end{aligned}
$$

yielding the desired approximation factor.

Initial sorting takes $O(n \log n)$ time. Step 3 takes linear time per iteration, for a total of $O(n^2)$. Step 6 takes linear time per iteration. For the exhaustive search, the maximum number of candidate piercing points (corner and intersection points) is $O(|R'_i|^2)$. The minimum number of points to pierce R'_i ($i \in \mathcal{I}$) can be at most $2kk'+k-1$, because $|R'_j| < 2kk'$ for $j \notin \mathcal{I}$ by Lemma 1, and incrementing j may only increase $|R'_j|$ by k. Checking whether a given set of $O(1)$ points is indeed a piercing set of S'_i trivially takes $O(|S'_i|)$ time. Therefore, step 7 takes $O(|S'_i|^{4kk'+2k-1})$ time for each $i \in \mathcal{I}$, for a total of $O(n^{4kk'+2k-1})$. □

Theorem 3 *For a set of n axis-parallel unit height rectangles and a constant $\epsilon > 0$, a factor-$(1+\epsilon)$ approximation of the piercing number can be computed in $n^{O(1/\epsilon^2)}$ time.*

Proof. The approximation algorithm uses the idea of shifting proposed by Hochbaum and Maass [9]:

1. Set $k' \leftarrow k \leftarrow \lceil 3/\epsilon \rceil$.
2. For $i \leftarrow 0$ to $k-1$ do
    3. Let $R^{(i,j)}$ be the subset of all rectangles stabbed by k consecutive horizontal lines $y = jk+i, jk+i+1, \ldots, (j+1)k+i-1$.
    4. Find a piercing set $P^{(i,j)}$ of each $R^{(i,j)}$ by Lemma 2, in total time $O(n^{4k^2+2k-1}) = n^{O(1/\epsilon^2)}$.
    5. Let $P^{(i)} = \bigcup_j P^{(i,j)}$.
6. Return the set P with minimum cardinality among $P^{(0)}, P^{(1)}, \ldots, P^{(k-1)}$.

Let Z denote the set of piercing points in an optimal solution. For an integer t, let $Z^{(t)}$ denote the subset of all points in Z inside the strip $t-1 \le y < t$. Since rectangles in $R^{(i,j)}$ are contained in $Z^{(jk+i)} \cup Z^{(jk+i+1)} \cup \cdots \cup Z^{((j+1)k+i)}$, and since Lemma 2 yields a $(1+1/k)$-factor approximation, we have $|P^{(i,j)}| \le (1+1/k)(|Z^{(jk+i)}| + |Z^{(jk+i+1)}| \cdots + |Z^{((j+1)k+i)}|)$. Therefore,

$$
|P^{(i)}| = \sum_j |P^{(i,j)}| \le \left(1+\frac{1}{k}\right)\left(|Z| + \sum_j |Z^{(jk+i)}|\right)
$$

(as the terms $|Z^{(jk+i)}|$ need to be double-counted). Thus,

$$
\begin{aligned}
|P| &\le \frac{1}{k}\sum_{i=0}^{k-1} |P^{(i)}| \\
&\le \left(1+\frac{1}{k}\right)\left(|Z| + \frac{1}{k}\sum_{i=0}^{k-1}\sum_j |Z^{(jk+i)}|\right) \\
&\le \left(1+\frac{1}{k}\right)\left(|Z| + \frac{1}{k}|Z|\right) \\
&= \left(1+\frac{1}{k}\right)^2 |Z| \;\le\; (1+\epsilon)|Z|.
\end{aligned}
$$

□

The approach can be extended to higher dimensions for axis-aligned boxes where all side lengths are equal to 1 except along one dimension.

3 An Incremental Factor-2 Algorithm

We now give a data structure that can maintain a factor-2 approximation of the piercing number for unit-height rectangles under insertions. By the method in Lemma 1, it suffices to give an insertion-only data structure for the exact 1-dimensional interval piercing problem.

Let $\mathcal{I} = \{I_1, \ldots, I_n\}$ be a set of n intervals with $I_k = [l_k, r_k]$, sorted by their right endpoints r_k. For convenience, set $I_0 = [-\infty, -\infty]$ and $I_\infty = [\infty, \infty]$. In the static instance, a very simple greedy approach finds the minimum piercing set: repeatedly print the leftmost right endpoint and remove all intervals pierced by this point. Notice that if r_i and r_j $(i < j)$ are two consecutive piercing points chosen by the greedy algorithm, then $r_j = \min_k\{r_k \mid l_k > r_i\}$. This motivates the following definition (used also in previous work such as [11]):

Definition 4 *Let* $NEXT(I_i) = I_j$ *if* $r_j = \min_k\{r_k \mid l_k > r_i\}$.

By forming a directed graph T with vertices $\mathcal{I} \cup \{I_0, I_\infty\}$ and edges $\{(I_i, I_j) \mid NEXT(I_i) = I_j\}$, the piercing number thus corresponds to the length of the path from I_0 to I_∞. Notice that T is acyclic with all vertices of out-degree 1 (excluding I_∞) and is thus a rooted tree.

To solve the dynamic piercing problem, we could maintain T in a data structure for *dynamic trees* [14] that supports path-length queries. However, the insertion of a single interval can cause as many as $\Omega(n)$ edge changes to T in the worst case. Thus, we will maintain a modified tree T'.

Before describing T', we first need two subroutines.

(a) Given any I_i, compute $NEXT(I_i)$:

We can determine $\min_k\{r_k \mid l_k > r_i\}$ in logarithmic time by using a *priority search tree* [12] to store the intervals I_k, ordered by l_k, with priorities defined by r_k; this data structure can be maintained in logarithmic time per update and can report the minimum priority among all elements in any query range.

(b) Given any I_j, identify all I_i's such that $NEXT(I_i) = I_j$ (the "reverse" of (a)):

Observe that $NEXT(I_i) = I_j$ iff $l_j > r_i$ and ($\forall k$, $l_k > r_i \Rightarrow r_j \leq r_k$), iff $l_j > r_i$ and ($\forall k$, $r_j > r_k \Rightarrow l_k \leq r_i$), iff $l_j > r_i \geq \max_k\{l_k \mid r_k < r_j\}$. All such I_i's thus appear in consecutive order (as intervals are sorted by right endpoints), and we can determine the first and last of these intervals in logarithmic time by another priority search tree, this time, with the intervals I_k ordered by r_k, and priorities defined by l_k.

Define a *block* to be a maximal set of intervals with a common $NEXT$ value. Blocks are obviously disjoint. The crucial observation, as shown in (b) above, is that elements within a block are consecutive. We form the new graph T' as follows. Vertices are intervals. We add an edge of weight 0 between every pair of consecutive vertices in the same block. We place an edge of weight 1 from I_i to $NEXT(I_i)$ only when i is the *last* vertex of a block. Clearly, T' is still a tree, and distances in T' are identical to distances in T, so the piercing number corresponds to the total weight of the path from I_0 to I_∞ in T'.

Sleator and Tarjan's original dynamic tree structure [14] supports edge insertions and deletions in T' (links and cuts) and certain queries (maximizing edge costs along a path) in logarithmic amortized time. For our particular type of queries (summing edge weights along a path), we can use Alstrup *et al.*'s top-tree implementation [1]. Insertion of a new interval I can trigger various changes in T'. There is at most one insertion of weight-1 edges leaving I (computable by (a)) and at most one insertion of weight-1 edges entering I (computable by (b)). In addition, there is at most one deletion of weight-0 edges caused by splitting of a block, as well as insertions of weight-0 edges and deletions of weight-1 edges caused by merging of blocks. In each interval insertion, the number of splits is bounded by a constant but the number of merges may be large. However, since the total number of merges is bounded by n plus the number of splits, the amortized number of edge changes is only $O(1)$. Furthermore, the location of the merges and splits can be identified in logarithmic time by having an extra balanced search tree holding the blocks' boundaries. The overall amortized time to maintain T' is thus $O(\log n)$. We conclude that:

Theorem 5 *In an insertion-only scenario, the piercing number of a set of n intervals can be maintained in $O(\log n)$ amortized time per insertion.*

Corollary 6 *In an insertion-only scenario, a factor-2 approximation of the piercing number of a set of n unit-height axis-parallel rectangles can be maintained in $O(\log n)$ amortized time per insertion.*

4 Open Questions

Our work raises two interesting questions:

1. Can we design an $n^{O(1/\epsilon)}$-time PTAS for axis-parallel unit-height rectangles? This can be answered if there is a polynomial-time exact algorithm for the special case considered in Lemma 2 (where the rectangles are stabbed by a constant number of horizontal lines).

2. Can the fully dynamic interval piercing problem be solved in $O(\log^{O(1)} n)$ time per insertion and deletion? Obtaining a near-$O(\sqrt{n})$ fully dynamic solution is not difficult (by partitioning the endpoints

into $\sqrt{n}$ groups and storing each group in a static data structure).

References

[1] S. Alstrup, J. Holm, K. de Lichtenberg, and M. Thorup. Maintaining information in fully-dynamic trees with top trees. http://arxiv.org/abs/cs/0310065, 2003.

[2] T. M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *Journal of Algorithms*, 46:178–189, 2003.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 2nd edition, 2001.

[4] A. Efrat, M. J. Katz, F. Nielsen, and M. Sharir. Dynamic data structures for fat objects and their applications. *Computational Geometry: Theory and Applications*, 15(4):215–227, April 2000.

[5] T. Feder and D. H. Greene. Optimal algorithms for approximate clustering. *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 434–444, 1988.

[6] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters*, 12(3):133–137, June 1981.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, New York, 1979.

[8] T. F. Gonzalez. Covering a set of points in multidimensional space. *Information Processing Letters*, 40:181–188, November 1991.

[9] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, January 1985.

[10] M. J. Katz, F. Nielsen, and M. Segal. Maintenance of a piercing set for intervals with applications. *Algorithmica*, 36(1):59–73, February 2003.

[11] S. Langerman. On the shooter location problem: Maintaining dynamic circular-arc graphs. In *Proceedings of the 12th Canadian Conference on Computational Geometry*, 2000.

[12] E. M. McCreight. Priority search trees. *SIAM Journal of Computing*, 14(2):257–276, May 1985.

[13] F. Nielsen. Fast stabbing of boxes in high dimensions. *Theoretical Computer Science*, 246:53–72, 2000.

[14] D. E. Sleator and R. E. Tarjan. A data structure for dynamic trees. *Journal of Computer and System Sciences*, 26(3):362–391, June 1983.