Dynamic Streaming Algorithms for Geometric Independent Set

Timothy M. Chan
Siebel School of Computing and Data Science, University of Illinois Urbana-Champaign, IL, USA

Yuancheng Yu
Siebel School of Computing and Data Science, University of Illinois Urbana-Champaign, IL, USA

Abstract

We present the first space-efficient, fully dynamic streaming algorithm for computing a constant-factor approximation of the maximum independent set size of n axis-aligned rectangles in two dimensions. For an arbitrarily small constant $\delta > 0$, our algorithm obtains an $O((1/\delta)^2)$ approximation and requires $O(U^\delta \operatorname{polylog} n)$ space and update time with high probability, assuming that coordinates are integers bounded by U. We also obtain a similar result for fat objects in any constant dimension. This extends recent non-streaming algorithms by Bhore and Chan from SODA'25, and also greatly extends previous streaming results, which were limited to special types of geometric objects such as one-dimensional intervals and unit disks.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Geometric Independent Set, Dynamic Streaming Algorithms

Digital Object Identifier 10.4230/LIPIcs.WADS.2025.17

1 Introduction

In this paper, we study dynamic streaming algorithms for a fundamental geometric optimization problem: *maximum independent set* (MIS) of geometric intersection graphs. Given a set S of n objects in d dimensions, the problem is to compute a maximum-cardinality subset $I \subseteq S$ of *independent* (i.e., pairwise-disjoint) objects. Exact geometric MIS is NP-hard even for unit squares[1] in 2D, and there has been a rich body of work on approximation algorithms [33, 25, 26, 16, 19, 27, 22]. The dynamic version of the problem (maintaining an approximate solution under insertions and deletions of objects in S) has also gained much attention in the last few years [32, 6, 10, 13, 8].

In particular, MIS for rectangles in 2D has been extensively studied. Early algorithms, based on simple divide-and-conquer, achieved logarithmic approximation ratios [2, 37, 41, 17, 5]. For $(1 + \varepsilon)$-approximation, Adamaszek, Har-Peled, and Weise [1] obtained a quasi-PTAS with running time $n^{\operatorname{polylog} n}$, which was subsequently improved to $n^{\operatorname{polyloglog} n}$ by Chuzhoy and Ene [21]. For polynomial-time results, Chalermsook and Chuzhoy [14] obtained an $O(\log \log n)$-approximation algorithm via LP rounding, and in a major breakthrough, Mitchell [39] obtained the first $O(1)$-approximation algorithm, running in $O(n^{21})$ time, via dynamic programming (DP). The approximation ratio was subsequently improved to $2 + \varepsilon$ for any constant $\varepsilon > 0$ by Gálvez, Khan, Mari, Mömke, Pittu, and Wiese [31, 30]. Recently, the running time has also been improved to $O(n^{1+\delta})$ for any constant $\delta > 0$ by Bhore and Chan [8], with approximation ratio dependent on $1/\delta$ (polynomially); if only an approximate size is desired, the time bound can be further improved to $O(n \operatorname{polylog} n)$. With the static time bound reduced to near linear, one can next turn to the question of efficient dynamic

[1] Throughout this paper, all squares and rectangles are axis-aligned by default.

algorithms. Indeed, Bhore and Chan showed that their static algorithm can be dynamized to obtain the first fully dynamic, constant-approximation algorithm, with $O(n^\delta)$ amortized update time.

For fat objects in any constant dimension d, PTASes were known [33, 16, 25], but the running time is high ($n^{\Omega(1/\varepsilon)}$). For faster algorithms, a simple greedy strategy gives an $O(1)$-approximation in quadratic time [25], and Bhore and Chan [7] recently obtained an improved $O(n \log n)$-time, $O(1)$-approximation algorithm. Bhore and Chan [8] also obtained fully dynamic $O(1)$-approximation algorithms with $O(n^\delta)$ amortized update time. For disks in $\mathbb{R}^2$ specifically, polylogarithmic amortized update time is also known [6, 7].

Meanwhile, *streaming* algorithms for geometric optimization problems also have a rich history. The dynamic streaming model, allowing streams of updates (both insertions and deletions), is also known as the *turnstile* model (viewing insertions/deletions as incrementing/ decrementing the multiplicity of an element – we assume that the multiplicity of each element is nonnegative). We seek algorithms that use one pass and small (sublinear) space. Typically, one needs to assume that input points lie in a bounded integer universe[2] $[U]^d$, and allow (Monte Carlo) randomization. For the streaming version of MIS, it is more natural to consider outputting an approximate size rather than the solution itself (since the MIS itself may have linear size).

Indyk [35] initiated the study of dynamic streaming algorithms for geometric problems, and considered basic problems such as Euclidean minimum spanning tree (EMST), min-weight matching, facility location, and k-median. Follow-up works investigated k-clustering [11, 29, 34], facility location [24], maximum cut [29, 38, 20], EMST [28], geometric Steiner forest [23], width and ε-kernels [3, 18], etc.

Streaming algorithms for geometric MIS, however, have received relatively limited success. Cabello and Pérez-Lantero [12] obtained tight approximation ratio for MIS of 1D intervals in insertion-only streams. Bakshi, Chepurko, and Woodruff [4] extended their result to fully dynamic (turnstile) streams, and also gave a dynamic streaming algorithm with polylogarithmic space for MIS of weighted unit disks in 2D. Bhore, Klute, and Oostveen [9] gave a 4-approximate insertion-only streaming algorithm for axis-aligned unit-height rectangles using $O(\mathrm{OPT} \operatorname{polylog} n)$ space, where OPT denotes the size of the MIS (which could be large). They also showed that for line segments, constant-approximation algorithms require linear space even for insertion-only streams. The above lines of work raise the following natural question:

"Do there exist fully dynamic streaming $O(1)$-approximation algorithms with sublinear space and update time for MIS for rectangles, or other types of geometric objects (more general than 1D intervals, 2D unit-height rectangles, and 2D unit disks)?"

New results. We answer this question in the affirmative by presenting the first space-efficient, fully dynamic streaming $O(1)$-approximation algorithm for MIS for rectangles in the plane. Specifically, our algorithm computes an $O((1/\delta)^2)$-approximation of the optimal size with $O(U^\delta \operatorname{polylog} n)$ space and update time w.h.p.,[3] where coordinates are in $[U]^2$. In particular, for polynomially bounded universe size $U = n^{O(1)}$, the space and update time is $O(n^\delta)$ (after adjusting δ by a constant factor).

[2] $[U]$ denotes $\{0, 1, \ldots, U-1\}$.
[3] With high probability, i.e., probability $1 - O(1/n^c)$ for an arbitrarily large constant c.

We also obtain a similar result for (unweighted and weighted) fat objects in any constant dimension d; in particular, this is new even for arbitrary-radii disks in the plane. Here, our algorithm computes an $O((1/\delta)^2)$-approximation of the optimal size with $O(U^\delta \operatorname{polylog} n)$ space and update time w.h.p., assuming that the objects are contained in $[0, U)^d$ and have diameter $\Omega(1)$ (and weights are in $[U]$).

Techniques. Our algorithms are based on Bhore and Chan's recent dynamic algorithms in the non-streaming setting [8]. Although most dynamic data structures do not adapt well to the streaming model and inherently require linear space or worse, we show that Bhore and Chan's approach does, luckily! Their approach for MIS for rectangles can be viewed as a combination of the old divide-and-conquer method [2] with Mitchell's constant-approximation DP-based algorithm [39]. More precisely, they used a b-way divide-and-conquer to reduce the problem to special subproblems in which the input rectangles are stabbable by $O(b)$ horizontal and vertical lines. The approximation ratio increases by an $O((\log_b n)^2)$ factor, which is constant if we choose $b = n^\delta$. They observed that for these special subproblems, one can round the input to reduce the input size to $b^{O(1)}$, while increasing the approximation ratio by at most a constant; when the input size is this small, one can afford to run Mitchell's polynomial-time approximation algorithm as a black box.

It turns out that the rounding trick is a perfect fit for dynamic streaming. The divide-and-conquer part, however, requires rethinking. We provide a new interpretation of the divide-and-conquer, by directly assigning input rectangles to levels of the recursion (this is possible due to the bounded integer universe assumption, with U^δ factors instead of n^δ); this simpler interpretation actually helps understand Bhore and Chan's algorithm better. To achieve sublinear space, we cannot afford to solve all subproblems across a level. Rather, we estimate the answer by taking a small random sample of the subproblems (the idea of using random sampling to approximate a sum appeared already in one version of Bhore and Chan's *static* algorithm [8], but not dynamic). In the dynamic setting, the right sampling rate is not known ahead of time. Our idea is to try all sampling rates (powers of 2) simultaneously. However, for sampling rates that are too large, the space usage would be too large. We resolve this issue by hashing to a small universe. This ensures that space is kept small for all sampling rates, and at the same time, for the right sampling rate, collision is likely avoided. We remark that sampling and hashing are both commonly used in the streaming literature, but we will keep our presentation largely self-contained, to make it accessible to computational geometers without prior background on streaming.

MIS for fat objects can be solved in a similar way, using shifted quadtrees for the divide-and-conquer part.

2 Preliminaries on Streaming

We begin by describing a key tool on dynamic streaming that we need. We formulate it in a general form, as stated in Theorem 1 below (even though the ideas in the proof may not be new to experts on streaming, our formulation will make applications easier).

The setup is roughly the following: we have a collection of subproblems, where each input object is assigned to one subproblem (a substream). Each subproblem individually admits an efficient dynamic streaming algorithm. We want to approximate the sum of the answers to these subproblems, subject to a stream of updates (insertions and deletions of objects), assuming that answers to nonempty subproblems lie in a bounded range $[1, B]$. Obtaining sublinear space is a challenge, because we cannot afford to explicitly store the answers to all subproblems.

▶ **Theorem 1.** *Let f be any function that maps sets to $\{0\} \cup [1, B]$ and satisfies the following two properties:*

- *$f(S) = 0$ iff $S = \emptyset$;*
- *there is a dynamic streaming algorithm for maintaining $f(S)$ under insertions and deletions of objects in S with $s(n)$ space and $t(n)$ time w.h.p., where $n = |S|$.*

Then we can design a dynamic streaming algorithm for maintaining a $(1 \pm \varepsilon)$-approximation of $T = \sum_{i=1}^{M} f(S_i)$, for a collection of sets $S_1, \ldots, S_M$, with $O(s(n)(B/\varepsilon)^{O(1)} \log^{O(1)}(nM))$ space and $O(t(n)(B/\varepsilon)^{O(1)} \log^{O(1)}(nM))$ time per update (insertion/deletion of an object to S_i for a given i), where $n = \sum_{i \in [M]} |S_i|$, w.h.p.

Before proving the theorem, we first consider a special case stated in the lemma below: when the number of nonempty subproblems is small, we can actually maintain the sum exactly. The precise statement is a little subtle, since the number of nonempty subproblems may vary over time as objects are inserted and deleted: at times when the number is larger than the threshold τ, the maintained answer may not be correct, but when the number gets back to below τ, the maintained answer would become correct again. The lemma is proved by a simple bucketing approach using pairwise-independent hash functions, with $O(\tau^2)$ buckets.

▶ **Lemma 2.** *Let f be any function that maps sets to numbers and satisfies the following two properties:*

- *$f(S) = 0$ iff $S = \emptyset$;*
- *there is a dynamic streaming algorithm $\mathcal{A}$ for maintaining $f(S)$ under insertions and deletions of objects in S with $s(n)$ space and $t(n)$ time w.h.p., where $n = |S|$.*

Let τ be a parameter. Then we can design a dynamic streaming algorithm for maintaining $\widetilde{T} = \sum_{i=1}^{M} f(S_i)$ for a collection of sets $S_1, \ldots, S_M$, with $O(s(n)\tau^2 \log n)$ space and $O(t(n)\tau^2 \log n)$ time per update (insertion/deletion of an object to S_i for a given i), where $n = \sum_{i=1}^{M} |S_i|$. The algorithm is correct w.h.p. when $|\{i \in [M] : S_i \neq \emptyset\}| < \tau$. Furthermore, this condition can be tested w.h.p. with the same space and time.

Proof. Let $h : [M] \to [q]$ be a random function chosen from a pairwise independent hash family, for $q = 4\tau^2$; by standard constructions (e.g. [40, Theorem 8.16]), h can be represented using $O(\log m)$ space.

Our streaming algorithm is simple: for each $a \in [q]$, we use the given algorithm $\mathcal{A}$ to maintain $f(Y_a)$ for the set

$$Y_a \;=\; \bigcup_{i \in [M]:\ h(i) = a} S_i,$$

i.e., when we insert/delete an object in S_j, we compute $a = h(j)$ and then insert/delete that object in Y_a using algorithm $\mathcal{A}$. Finally, let

$$y := |\{a \in [q] : Y_a \neq \emptyset\}|,$$

and when $y < \tau$, we output

$$\widehat{T} \;=\; \sum_{a \in [q]} f(Y_a).$$

For correctness, we discuss the two cases. Suppose $|\{i \in [M] : S_i \neq \emptyset\}| < \tau$. We claim that $\widehat{T} = \widetilde{T}$ (and $y < \tau$ trivially) with probability at least $3/4$. To see this, observe that $\widehat{T} = \widetilde{T}$ when there are no *colliding pairs*, i.e., no pairs (i, j) with $i \neq j$, $S_i, S_j \neq \emptyset$, and $h(i) = h(j)$. The expected number of colliding pairs is at most $|\{i \in [M] : S_i \neq \emptyset\}|^2 \cdot 1/q < 1/4$. By Markov's inequality, the probability that there is a colliding pair is less than $1/4$.

Conversely, suppose $|\{i \in [M] : S_i \neq \emptyset\}| \geq \tau$. We claim that $y \geq \tau$ with probability at least $3/4$. To see this, pick τ nonempty subsets S_i. By the same argument as above, with probability at least $3/4$, there are no colliding pairs among these τ nonempty subsets S_i, implying $y \geq \tau$.

The error probability is bounded by a constant, but can be lowered by running logarithmically many independent copies of the algorithm and taking the majority of the answers. ◀

Note that alternatively, the above condition $|\{i \in [M] : S_i \neq \emptyset\}| < \tau$ can be tested by directly applying known techniques for sampling elements (an "ℓ_0-sampler") in a turnstile stream [28, 36] (viewing an insertion/deletion in S_i as incrementing/decrementing the multiplicity of $i \in [M]$). The proof above may be viewed as an adaptation or extension of such techniques.

We now reduce the general case to the special case by random sampling, using logarithmically many random subsets with different sampling rates. Chebyshev's inequality shows that the sum of a random subset approximates the overall sum, assuming pairwise independence.

▶ **Lemma 3.** *Let $T = \sum_{i=1}^m a_i$, where $a_i \in [1, B]$. Let R be a random subset of $[m]$ such that for each $i \in [m]$, $\Pr[i \in R] = \rho$, and the events "$i \in R$" over all $i \in [m]$ are pairwise independent. For $\varepsilon < 1$, if $\rho m \geq cB/\varepsilon^2$, then $(1/\rho)\sum_{i\in R} a_i$ gives a $(1 \pm \varepsilon)$-approximation of T with probability $1 - O(1/c)$.*

Proof. Let $X = \sum_{i\in R} a_i = \sum_{i=1}^m a_i \cdot [i \in R]$, where $[\cdot]$ denotes the Iverson bracket. Then $\mathbb{E}[X] = \rho T$, and by pairwise independence, $\mathrm{Var}[X] = \sum_{i=1}^m a_i^2 \rho \leq B \sum_{i=1} a_i \rho = B\rho T$. By Chebyshev's inequality, $\Pr[|X - \mathbb{E}[X]| \geq \varepsilon \mathbb{E}[X]] \leq \mathrm{Var}[X]/(\varepsilon^2 \mathbb{E}[X]^2) \leq B/(\varepsilon^2 \rho T) \leq B/(\varepsilon^2 \rho m) \leq 1/c$. ◀

Proof of Theorem 1. For each $\rho = 1/2, 1/4, \ldots, 1/2^{\lceil \log M \rceil}$, let R_ρ be a random subset of $[M]$ with sampling rate ρ, where the events "$i \in R_\rho$" are pairwise independent. By standard constructions (e.g. [40, Theorem 8.16]), such a subset can be represented using $O(\log m)$ space (so that we can tell if a given index i is in R_ρ in $O(1)$ time).

We will apply the algorithm in Lemma 2 to try to maintain

$$\widetilde{T}_\rho = \sum_{i \in R_\rho} f(S_i),$$

with $\tau = cB/\varepsilon^2$ for a sufficiently large constant c. Pick the largest ρ^* for which $|\{i \in R_{\rho^*} : S_i \neq \emptyset\}| < \tau$ (recall that this condition can be tested w.h.p. by Lemma 2). Then the algorithm from Lemma 2 will succeed in computing $\widetilde{T}_{\rho^*}$ w.h.p. and we return $(1/\rho^*)\widetilde{T}_{\rho^*}$.

For the analysis, let $m = |\{i \in [M] : S_i \neq \emptyset\}|$. Let ρ_0 be a power of 2 with $\tau/4 < \rho_0 m \leq \tau/2$. Then by Chebyshev's inequality,

$$\Pr[\rho^* < \rho_0] \leq \Pr[|\{i \in R_{\rho_0} : S_i \neq \emptyset\}| \geq \tau] \leq \rho_0 m/(\tau - \rho_0 m)^2 \leq O(1/(\rho_0 m)).$$

On the other hand, for each $\rho \geq 8\rho_0$, by Chebyshev's inequality,

$$\Pr[\rho^* = \rho] \leq \Pr[|\{i \in R_\rho : S_i \neq \emptyset\}| < \tau] \leq \rho m/(\rho m - \tau)^2 \leq O(1/(\rho m)).$$

By a geometric series over all $\rho \geq 8\rho_0$, $\Pr[\rho^* \geq 8\rho_0] \leq O(1/(\rho_0 m))$. Thus, with probability $1 - O(1/c)$, ρ^* must be a value among $\{\rho_0, 2\rho_0, 4\rho_0\}$. For each such value ρ, the probability that $\rho^* = \rho$ but $(1/\rho)\widetilde{T}_\rho$ is not a $(1 \pm \varepsilon)$-approximation of T is at most $O(1/c)$, by applying Lemma 3 to the sum $T = \sum_{i\in[M]:S_i \neq \emptyset} f(S_i)$, which has m nonempty terms.

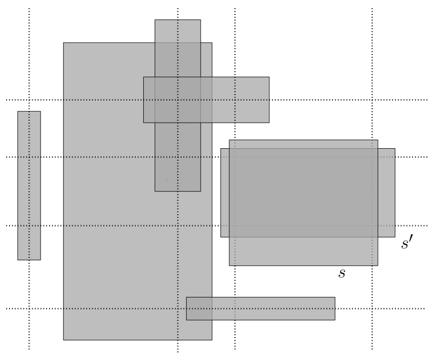

Figure 1 Proof of Lemma 4. Each rectangle is stabbed by at least one horizontal line and at least one vertical line of Γ (shown as dotted lines). The rectangles s and s' belong to the same class.

The error probability is bounded by a constant, but can again be lowered by running logarithmically many independent copies of the algorithm and taking the majority of the answers. (For simplicity, we have not attempted to optimize the $(B/\varepsilon)^{O(1)} \log^{O(1)}(nM)$ factors.) ◀

3 Maximum Independent Set for Rectangles

We now present our dynamic streaming algorithm for MIS for rectangles. We begin by considering a special case in which the input rectangles are stabbable by a small number of horizontal as well as vertical lines. Bhore and Chan [8] solved this case by a simple rounding trick, which we observe works well in the dynamic streaming setting:

▶ **Lemma 4.** *Let Γ be a set of $O(b)$ horizontal/vertical lines in 2D. Let S be a set of n axis-aligned rectangles in 2D with the property that each rectangle in S is stabbed by at least one horizontal line and at least one vertical line in Γ. Then we can design a dynamic streaming algorithm for maintaining an $O(1)$-approximation of the maximum independent set for S with $O(b^{O(1)} \log^{O(1)} n)$ space and time w.h.p.*

Proof. The lines in Γ form a (non-uniform) grid with $O(b^2)$ grid cells. (See Figure 1.) Place two rectangles of S in the same *class* if they intersect the same subset of grid cells. There are $O(b^4)$ classes. Let $\hat{S}$ be a subset of S where we keep one "representative" element from each class. Then $|\hat{S}| = O(b^4)$. We apply Mitchell's algorithm [39] (or its subsequent improvement [31, 30]) to compute an $O(1)$-approximation of the maximum independent set for the rectangles in $\hat{S}$. This takes time polynomial in $|\hat{S}|$, i.e., $b^{O(1)}$ time (note that the optimal size must be at most b^2 under the input assumption). Bhore and Chan [8] proved that this is an $O(1)$-approximation of the maximum independent set for S.

To implement the algorithm in the dynamic streaming model, it suffices to maintain one representative element per class, since we can re-run Mitchell's algorithm from scratch. Frahling, Indyk, and Sohler [28] (see also [36]) showed how to maintain a random element of a set in the dynamic streaming model; we can just apply their algorithm to each class and use the random element as the representative. ◀

We now solve the main problem by reinterpreting the divide-and-conquer approach by Bhore and Chan [8] and incorporating Theorem 1.

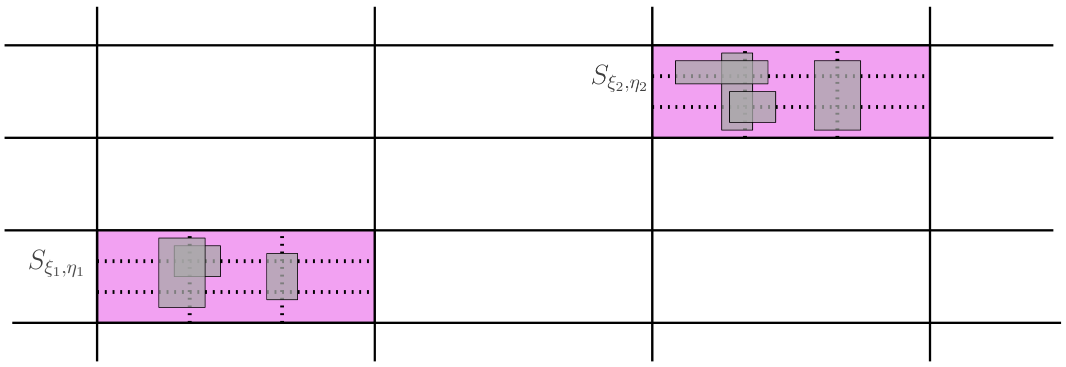

Figure 2 Proof of Theorem 5. The picture shows two different sets S_{ξ_1,η_1} and S_{ξ_2,η_2} from a common collection $S_{k,\ell}$.

▶ **Theorem 5.** *Given a set S of n axis-aligned rectangles in 2D with coordinates in $[U]$ as a dynamic geometric stream, we can compute an $O((1/\delta)^2)$-approximation of the maximum independent set size for S using $O(U^{O(\delta)}\log^{O(1)} n)$ space and update time w.h.p.*

Proof. Write each coordinate value in base b; the number of digits in each value is $\lceil \log_b U\rceil$.

Consider a rectangle $s = [x, x'] \times [y, y']$ in S. Suppose that the most significant digit which x and x' differ in is the k-th most significant digit, and the most significant digit which y and y' differ in is the ℓ-th most significant digit. Let ξ be the $k-1$ most significant digits of x and x'. Let η be the $\ell-1$ most significant digits of y and y'. We place s in a set $S_{\xi,\eta}$; in turn, we place the set $S_{\xi,\eta}$ in a collection $\mathcal{S}_{k,\ell}$.[4] (See Figure 2.)

For each $k, \ell \in [\lceil \log_b U\rceil]$, we maintain an $O(1)$-approximation of

$$T_{k,\ell} = \sum_{S_{\xi,\eta}\in\mathcal{S}_{k,\ell}} f_{k,\ell}(S_{\xi,\eta})$$

using Theorem 1. Here, $f_{k,\ell}(S)$ is defined as the maximum independent set size for $S^{\#} = \{s^{\#} : s \in S\}$, where $s^{\#}$ denotes the rectangle obtained from $s = [x, x'] \times [y, y']$ by removing the $k-1$ most significant digits from x and x' and the $\ell-1$ most significant digits from y and y', assuming that x and x' differ in the k-th most significant digit and y and y' differ in the ℓ-th most significant digit (if the assumption is not satisfied, make $s^{\#}$ undefined). Note that for an individual set $S_{\xi,\eta} \in \mathcal{S}_{k,\ell}$, $f(S_{\xi,\eta})$ is the same as the maximum independent set size for $S_{\xi,\eta}$ (although this may not be true for a general set S). Also note that every rectangle $s^{\#}$ is stabbed by one of the b vertical lines at $x = b^{\lceil \log_b U\rceil - k} i$ for some $i \in [b]$, and one of the b horizontal lines at $y = b^{\lceil \log_b U\rceil-\ell} j$ for some $j \in [b]$, and so we can maintain $f_{k,\ell}(S)$ using Lemma 4 (note that $f_{k,\ell}(S) \le b^2$). The conditions of Theorem 1 are fulfilled with $B = b^2$ and $s(n), t(n) = O(b^{O(1)}\log^{O(1)} n)$. We return the maximum of the approximations of $T_{k,\ell}$ over all $k, \ell \in [\lceil \log_b U\rceil]$.

Observe that $T_{k,\ell}$ is just the maximum independent set size for $\bigcup_{S_{\xi,\eta}\in\mathcal{S}_{k,\ell}} S_{\xi,\eta}$, because rectangles in S_{ξ_1,η_1} do not overlap with rectangles in S_{ξ_2,η_2} for any two different sets $S_{\xi_1,\eta_1}, S_{\xi_2,\eta_2} \in \mathcal{S}_{k,\ell}$. Thus, OPT (the maximum independent set size for S) is at most $\sum_{k,\ell} T_{k,\ell} \le O(\log_b U)^2 \cdot \max_{k,\ell} T_{k,\ell}$. It follows that the returned value is an $O((\log_b U)^2)$-approximation of OPT. Finally, we set $b = U^{\delta}$. ◀

[4] Roughly, Bhore and Chan's recursion [8] generates a primary tree (essentially a b-ary interval tree in x), where each node stores a secondary tree (a b-ary interval tree in y). Each set $S_{\xi,\eta}$ corresponds to a node of a secondary tree. Each collection $\mathcal{S}_{k,\ell}$ corresponds to a "level" of the whole structure (namely, the ℓ-th level of all nodes at the k-th level of the primary tree).

Although the preceding proof is compact, the details are subtle. For example, some readers may wonder why we didn't define $f_{k,\ell}(S)$ more simply as the maximum independent set size for S, bypassing the definition of $s^{\#}$. The reason is that Theorem 1 requires working with $f_{k,\ell}(S)$ for general sets S, not just the $S_{\xi,\eta}$'s: indeed, the proof of Lemma 2 maintains intermediate sets that are *unions* of $S_{\xi,\eta}$'s (from a common bucket); without replacing s with $s^{\#}$, the rectangles in such composite sets may not be stabbable by b vertical/horizontal lines. Admittedly, defining $f_{k,\ell}(S)$ as the maximum independent set for $S^{\#}$ is somewhat counterintuitive, as its value seems irrelevant or meaningless when S is a union of two or more sets $S_{\xi,\eta}$'s. (To picture such a set $S^{\#}$, imagine overlaying the two violet cells in Figure 2 on top of each other.) However, under the "right" circumstances, when we don't have collisions, as in the proof of Lemma 2, S would revert to a single $S_{\xi,\eta}$, and $f_{k,\ell}(S)$ would become meaningful again.

▶ Remark 6. Curiously, the approach of Theorem 5 extends to yield a space-efficient, $O(1)$-approximation streaming algorithm of boxes (hyperrectangles) in any constant dimension beyond 2. The catch is that the update time is prohibitively large ($2^{U^{O(\delta)}}$), as Mitchell's algorithm works only in 2D. (Obtaining a polynomial-time $O(1)$-approximation algorithm for boxes in 3D is a major open problem even in the non-streaming setting.)

As in Bhore and Chan's paper [8], our approach can also solve the *minimum piercing set* problem for 2D rectangles in the dynamic streaming model, but the approximation ratio increases to $O(\log\log n)$ with $O(U^{O(1)}\log n)$ space and update time.

4 Maximum Independent Set for Fat Objects

We now consider the case of fat objects, again adapting Bhore and Chan's non-streaming algorithm [8]. Our algorithm extends to the weighted version of independent set.

In the following, we adapt notation and definitions from [8]. Let $\mathrm{diam}(s)$ denote the diameter of an object s, where diameter refers to L_∞-diameter. A collection $\mathcal{C}$ of objects is *c-fat* if for every hypercube γ, there exist c points piercing all objects in $\mathcal{C}$ that intersect γ and have diameter at least $\mathrm{diam}(\gamma)$. A *quadtree box* γ is a hypercube of the form $[i_1 2^\ell, (i_1+1)2^\ell) \times \cdots \times [i_d 2^\ell, (i_d+1)2^\ell)$ for integers $i_1, \ldots, i_d, \ell$.

An object s is c_0*-good* if it is contained in a quadtree box with diameter at most $c_0\mathrm{diam}(s)$.

▶ **Fact 7** (Shifting Lemma [15, 16]). *Suppose d is even. Let $v_j = (\frac{jU}{d+1}, \ldots, \frac{jU}{d+1}) \in \mathbb{R}^d$. For every object $s \subset [0,U)^d$, there exists $j \in \{0, \ldots, d\}$ such that $s + v_j$ is $O(d)$-good.*

Bhore and Chan [8] solved the following special case by rounding, which we observe works well in the dynamic streaming setting:

▶ **Lemma 8.** *Let d, c, c_0 be constants. Let Γ be b disjoint quadtree boxes. Let S be a set of n c_0-good objects in $\mathbb{R}^d$ of constant description complexity from a c-fat collection $\mathcal{C}$, with the property that each object in S intersects the boundary of at least one quadtree box of Γ. Each object is given a weight in $[U]$. Then we can design a dynamic streaming algorithm for maintaining an $O(1)$-approximation of the maximum-weight independent set for S with $O(b^{O(1)}\log^{O(1)}(nU))$ space and time w.h.p.*

Proof. First round the weights to powers of 2; the approximation ratio increases by at most a factor of 2.

Place two objects of S in the same *class* if they intersect the same subset of cells in Γ. There are at most $b^{O(1)}$ classes (as noted in [8]). Let $\hat{S}$ be a subset of S where we keep one "representative" element from each class, namely, a largest-weight element from the class.

Then $|\hat{S}| \le b^{O(1)}$. We apply a known algorithm (e.g., [16]) to compute an $O(1)$-approximation to the maximum-weight independent set for the fat objects in $\hat{S}$. This takes time polynomial in $|\hat{S}|$, i.e., $b^{O(1)}$ time. Bhore and Chan [8] proved that this is an $O(1)$-approximation of the maximum independent set for S (using the goodness assumption).

To implement the algorithm in the dynamic streaming model, it suffices to maintain a representative element per class. Frahling, Indyk, and Sohler [28] (see also [36]) showed how to maintain a random element of a set in the dynamic streaming model; we can just apply their algorithm to each class restricted to elements of each weight, and use the random element as the representative for the largest weight whose set is nonempty. There are only $O(\log U)$ possible weights. ◀

We now solve the main problem, via quadtree-based divide-and-conquer, as in [8],[5] but in combination with Theorem 1.

▶ **Theorem 9.** *For any constant dimension d, given a set S of n c-fat objects in $[0, U)^d$ as a dynamic geometric stream, where the diameter of each object is at least 1 and each object has a weight in $[U]$, we can compute an $O((1/\delta)^2)$-approximation of the maximum independent set weight for S using $O(U^{O(\delta)} \log^{O(1)} n)$ space and update time w.h.p.*

Proof. We assume that all objects of S are $O(d)$-good. This is without loss of generality by the shifting lemma (Fact 7): for each of the $d+1$ shifts v_j ($j \in \{0, \ldots, d\}$), we can solve the problem for the good objects of $S + v_j$ in parallel, and return the maximum of the answers. The approximation ratio increases by a factor of $d + 1 = O(1)$.

Furthermore, we assume that all objects have weights in a common interval $[b^k, b^{k+1})$. This is without loss of generality, if we increase the approximation ratio by an extra factor of $O(\log_b U)$ (since there are $O(\log_b U)$ choices for k). By rescaling, we may now assume that all objects have weights in $[1, b)$.

Consider an object $s \in S$. Let γ be the smallest quadtree box containing s such that $\mathrm{diam}(\gamma)$ is a power of b (the parameter b will be a power of 2). Suppose that $\mathrm{diam}(\gamma) = b^\ell$. We place s in a set S_γ; in turn, we place the set S_γ in a collection $\mathcal{S}_\ell$.[6]

For each $\ell \in [\log_b U]$, we maintain an $O(1)$-approximation of $T_\ell = \sum_{S_\gamma \in \mathcal{S}_\ell} f_\ell(S_\gamma)$ using Theorem 1. Here, $f_\ell(S)$ is defined as the maximum independent set weight for $S^\# = \{s^\# : s \in S\}$, where $s^\#$ denotes the object s shifted by $-p_\gamma$, where γ is the quadtree box containing s with $\mathrm{diam}(\gamma) = b^\ell$, and p_γ is the lexicograpically smallest vertex of γ, assuming that s is not contained in any quadtree box with diameter $b^{\ell-1}$ (if the assumption is not satisfied, make $s^\#$ undefined). Note that for an individual set $S_\gamma \in \mathcal{S}_\ell$, $f(S_\gamma)$ is the same as the maximum independent set weight for S_ℓ. Also note that every object $s^\#$ intersects the boundary of one of the b^d quadtree boxes of diameter $b^{\ell-1}$ contained in $[0, b^\ell)^d$, and so we can maintain $f_\ell(S)$ using Lemma 4 after replacing b with b^d (note that $f_\ell(S) \le b^{O(1)}$, since all objects have weights in $[1, b)$). The conditions of Theorem 1 are fulfilled with $B = b^{O(1)}$ and $s(n), t(n) = O(b^{O(1)} \log^{O(1)}(nU))$. We return the maximum of the approximations of T_ℓ over all $\ell \in [\log_b U]$.

Observe that T_ℓ is just the maximum independent set weight for $\bigcup_{S_\gamma \in \mathcal{S}_\ell} S_\gamma$, because rectangles in S_{γ_1} do not overlap with rectangles in S_{γ_2} for any two different sets $S_{\gamma_1}, S_{\gamma_2} \in \mathcal{S}_\ell$. Thus, OPT (the maximum independent set weight for S) is at most $\sum_\ell T_\ell \le O(\log_b U) \cdot$

[5] Bhore and Chan [8] needed a balanced version of quadtrees (requiring cells that are differences of two quadtree boxes), but we are able to simplify the divide-and-conquer because of the bounded universe assumption.

[6] Each S_γ corresponds to a node in the b-ary variant of the quadtree. Each $\mathcal{S}_\ell$ corresponds to one level of the quadtree.

$\max_\ell T_\ell$. It follows that the returned value is an $O(\log_b U)$-approximation of OPT, under the assumption that objects have weights in $[1, b)$. We thus obtain approximation ratio $O((\log_b U)^2)$ for the general problem. Finally, we set $b = U^\delta$. ◀

▶ Remark 10. Without weights, the approximation ratio is $O(1/\delta)$. If the ratio of the maximum to the minimum diameter of the objects and the ratio of the maximum to the minimum weight of the objects are bounded by Δ, then the U^δ factors can be reduced to Δ^δ.

As in Bhore and Chan's paper [8], the approach also works for the minimum piercing set problem for fat objects.

References

1 Anna Adamaszek, Sariel Har-Peled, and Andreas Wiese. Approximation schemes for independent set and sparse subsets of polygons. *J. ACM*, 66(4):29:1–29:40, 2019. doi:10.1145/3326122.

2 Pankaj K. Agarwal, Marc J. van Kreveld, and Subhash Suri. Label placement by maximum independent set in rectangles. *Comput. Geom.*, 11(3-4):209–218, 1998. doi:10.1016/S0925-7721(98)00028-5.

3 Alexandr Andoni and Huy L. Nguyên. Width of points in the streaming model. *ACM Trans. Algorithms*, 12(1):5:1–5:10, 2016. doi:10.1145/2847259.

4 Ainesh Bakshi, Nadiia Chepurko, and David P. Woodruff. Weighted maximum independent set of geometric objects in turnstile streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, volume 176 of *LIPIcs*, pages 64:1–64:22, 2020. doi:10.4230/LIPICS.APPROX/RANDOM.2020.64.

5 Piotr Berman, Bhaskar DasGupta, S. Muthukrishnan, and Suneeta Ramaswami. Improved approximation algorithms for rectangle tiling and packing. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 427–436, 2001. URL: http://dl.acm.org/citation.cfm?id=365411.365496.

6 Sujoy Bhore, Jean Cardinal, John Iacono, and Grigorios Koumoutsos. Dynamic geometric independent set. In *Abstracts of 23rd Thailand-Japan Conference on Discrete and Computational Geometry, Graphs, and Games (TJDCG)*, 2021. arXiv:2007.08643.

7 Sujoy Bhore and Timothy M. Chan. Dynamic independent set of disks (and hypercubes) made easier. In *Proceedings of the 8th SIAM Symposium on Simplicity in Algorithms (SOSA)*, pages 485–495, 2025. doi:10.1137/1.9781611978315.36.

8 Sujoy Bhore and Timothy M. Chan. Fast static and dynamic approximation algorithms for geometric optimization problems: Piercing, independent set, vertex cover, and matching. In *Proceedings of the 36th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2357–2386, 2025. doi:10.1137/1.9781611978322.79.

9 Sujoy Bhore, Fabian Klute, and Jelle J. Oostveen. On streaming algorithms for geometric independent set and clique. In *Proceedings of the 20th International Workshop on Approximation and Online Algorithms (WAOA)*, volume 13538 of *Lecture Notes in Computer Science*, pages 211–224. Springer, 2022. doi:10.1007/978-3-031-18367-6_11.

10 Sujoy Bhore, Martin Nöllenburg, Csaba D. Tóth, and Jules Wulms. Fully dynamic maximum independent sets of disks in polylogarithmic update time. In *Proceedings of the 40th International Symposium on Computational Geometry (SoCG)*, volume 293 of *LIPIcs*, pages 19:1–19:16, 2024. doi:10.4230/LIPICS.SOCG.2024.19.

11 Vladimir Braverman, Gereon Frahling, Harry Lang, Christian Sohler, and Lin F. Yang. Clustering high dimensional dynamic data streams. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 576–585, 2017. URL: http://proceedings.mlr.press/v70/braverman17a.html.

12 Sergio Cabello and Pablo Pérez-Lantero. Interval selection in the streaming model. *Theor. Comput. Sci.*, 702:77–96, 2017. doi:10.1016/J.TCS.2017.08.015.

13 Jean Cardinal, John Iacono, and Grigorios Koumoutsos. Worst-case efficient dynamic geometric independent set. In *Proceedings of the 29th Annual European Symposium on Algorithms (ESA)*, volume 204 of *LIPIcs*, pages 25:1–25:15, 2021. `doi:10.4230/LIPICS.ESA.2021.25`.

14 Parinya Chalermsook and Julia Chuzhoy. Maximum independent set of rectangles. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 892–901. SIAM, 2009. `doi:10.1137/1.9781611973068.97`.

15 Timothy M. Chan. Approximate nearest neighbor queries revisited. *Discret. Comput. Geom.*, 20(3):359–373, 1998. `doi:10.1007/PL00009390`.

16 Timothy M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *J. Algorithms*, 46(2):178–189, 2003. `doi:10.1016/S0196-6774(02)00294-8`.

17 Timothy M. Chan. A note on maximum independent sets in rectangle intersection graphs. *Inf. Process. Lett.*, 89(1):19–23, 2004. `doi:10.1016/J.IPL.2003.09.019`.

18 Timothy M. Chan. Dynamic streaming algorithms for epsilon-kernels. In *Proceedings of the 32nd International Symposium on Computational Geometry (SoCG)*, volume 51 of *LIPIcs*, pages 27:1–27:11, 2016. `doi:10.4230/LIPICS.SOCG.2016.27`.

19 Timothy M. Chan and Sariel Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. *Discret. Comput. Geom.*, 48(2):373–392, 2012. `doi:10.1007/S00454-012-9417-5`.

20 Xiaoyu Chen, Shaofeng H.-C. Jiang, and Robert Krauthgamer. Streaming Euclidean max-cut: Dimension vs data reduction. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*, pages 170–182, 2023. `doi:10.1145/3564246.3585170`.

21 Julia Chuzhoy and Alina Ene. On approximating maximum independent set of rectangles. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 820–829, 2016. `doi:10.1109/FOCS.2016.92`.

22 Jana Cslovjecsek, Michał Pilipczuk, and Karol Wegrzycki. A polynomial-time OPT$^{\varepsilon}$-approximation algorithm for maximum independent set of connected subgraphs in a planar graph. In *Proceedings of the 35th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 625–638, 2024. `doi:10.1137/1.9781611977912.23`.

23 Artur Czumaj, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Pavel Veselý. Streaming algorithms for geometric Steiner forest. *ACM Trans. Algorithms*, 20(4):28:1–28:38, 2024. `doi:10.1145/3663666`.

24 Artur Czumaj, Christiane Lammersen, Morteza Monemizadeh, and Christian Sohler. $(1+\epsilon)$ -approximation for facility location in data streams. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1710–1728, 2013. `doi:10.1137/1.9781611973105.123`.

25 Alon Efrat, Matthew J. Katz, Frank Nielsen, and Micha Sharir. Dynamic data structures for fat objects and their applications. *Comput. Geom.*, 15(4):215–227, 2000. `doi:10.1016/S0925-7721(99)00059-0`.

26 Thomas Erlebach, Klaus Jansen, and Eike Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM Journal on Computing*, 34(6):1302–1323, 2005. `doi:10.1137/S0097539702402676`.

27 Jacob Fox and János Pach. Computing the independence number of intersection graphs. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1161–1165, 2011. `doi:10.1137/1.9781611973082.87`.

28 Gereon Frahling, Piotr Indyk, and Christian Sohler. Sampling in dynamic data streams and applications. *Int. J. Comput. Geom. Appl.*, 18(1/2):3–28, 2008. `doi:10.1142/S0218195908002520`.

29 Gereon Frahling and Christian Sohler. Coresets in dynamic geometric data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 209–217. ACM, 2005. `doi:10.1145/1060590.1060622`.

30 Waldo Gálvez, Arindam Khan, Mathieu Mari, Tobias Mömke, Madhusudhan Reddy Pittu, and Andreas Wiese. A $(2+\varepsilon)$-approximation algorithm for maximum independent set of rectangles. *CoRR*, abs/2106.00623, 2021. arXiv:2106.00623.

31 Waldo Gálvez, Arindam Khan, Mathieu Mari, Tobias Mömke, Madhusudhan Reddy Pittu, and Andreas Wiese. A 3-approximation algorithm for maximum independent set of rectangles. In *Proceedings of the 33rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 894–905, 2022. doi:10.1137/1.9781611977073.38.

32 Monika Henzinger, Stefan Neumann, and Andreas Wiese. Dynamic approximate maximum independent set of intervals, hypercubes and hyperrectangles. In *Proceedings of the 36th International Symposium on Computational Geometry (SoCG)*, volume 164 of *LIPIcs*, pages 51:1–51:14, 2020. doi:10.4230/LIPICS.SOCG.2020.51.

33 Dorit S. Hochbaum and Wolfgang Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, 1985. doi:10.1145/2455.214106.

34 Wei Hu, Zhao Song, Lin F Yang, and Peilin Zhong. Nearly optimal dynamic k-means clustering for high-dimensional data. *arXiv preprint*, 2018. arXiv:1802.00459.

35 Piotr Indyk. Algorithms for dynamic geometric problems over data streams. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 373–380, 2004. doi:10.1145/1007352.1007413.

36 Hossein Jowhari, Mert Saglam, and Gábor Tardos. Tight bounds for L_p samplers, finding duplicates in streams, and related problems. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 49–58, 2011. doi:10.1145/1989284.1989289.

37 Sanjeev Khanna, S. Muthukrishnan, and Mike Paterson. On approximating rectangle tiling and packing. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 384–393, 1998. URL: http://dl.acm.org/citation.cfm?id=314613.314768.

38 Christiane Lammersen, Anastasios Sidiropoulos, and Christian Sohler. Streaming embeddings with slack. In *Proceedings of the 11th International Symposium on Algorithms and Data Structures (WADS)*, volume 5664 of *Lecture Notes in Computer Science*, pages 483–494. Springer, 2009. doi:10.1007/978-3-642-03367-4_42.

39 Joseph S. B. Mitchell. Approximating maximum independent set for rectangles in the plane. In *Proceedings of the 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 339–350, 2022. doi:10.1109/FOCS52979.2021.00042.

40 Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995. doi:10.1017/CBO9780511814075.

41 Frank Nielsen. Fast stabbing of boxes in high dimensions. *Theor. Comput. Sci.*, 246(1-2):53–72, 2000. doi:10.1016/S0304-3975(98)00336-3.