On Streaming Algorithms for Geometric Independent Set and Clique

Sujoy Bhore[1], Fabian Klute[2], and Jelle J. Oostveen[2(✉)]

[1] Indian Institute of Science Education and Research, Bhopal, India
[2] Utrecht University, Utrecht, The Netherlands
{f.m.klute,j.j.oostveen}@uu.nl

Abstract. We study the maximum geometric independent set and clique problems in the streaming model. Given a collection of geometric objects arriving in an insertion only stream, the aim is to find a subset such that all objects in the subset are pairwise disjoint or intersect respectively.

We show that no constant factor approximation algorithm exists to find a maximum set of independent segments or 2-intervals without using a linear number of bits. Interestingly, our proof only requires a set of segments whose intersection graph is also an interval graph. This reveals an interesting discrepancy between segments and intervals as there does exist a 2-approximation for finding an independent set of intervals that uses only $O(\alpha(\mathcal{I}) \log |\mathcal{I}|)$ bits of memory for a set of intervals $\mathcal{I}$ with $\alpha(\mathcal{I})$ being the size of the largest independent set of $\mathcal{I}$. On the flipside we show that for the geometric clique problem there is no constant-factor approximation algorithm using less than a linear number of bits even for unit intervals. On the positive side we show that the maximum geometric independent set in a set of axis-aligned unit-height rectangles can be 4-approximated using only $O(\alpha(\mathcal{R}) \log |\mathcal{R}|)$ bits.

Keywords: Geometric independent set · Streaming algorithms · Geometric intersection graphs · Communication lower bounds

1 Introduction

The independent set problem is one of the fundamental combinatorial optimization problems in theoretical computer science, with a wide range of applications. Given a graph $G = (V, E)$, a set of vertices $M \subset V$ is *independent* if no two vertices in M are adjacent in G. A *maximum independent set* is a maximum cardinality independent set. Maximum independent set is one of the most well-studied algorithmic problems and is one of Karp's 21 classic NP-complete problems [33]. Moreover, it is well-known to be hard to approximate: no polynomial time algorithm can achieve an approximation factor $n^{1-\epsilon}$, for any constant $\epsilon > 0$,

Fabian Klute supported by the Austrian Science Foundation (FWF) grant J4510. Jelle Oostveen is partially supported by the NWO grant OCENW.KLEIN.114 (PACAN).

P. Chalermsook and B. Laekhanukit (Eds.): WAOA 2022, LNCS 13538, pp. 211–224, 2022.
https://doi.org/10.1007/978-3-031-18367-6_11

unless P = NP [28,39]. Maximum independent set serves as a natural model for many real-life optimization problems, including map labeling, computer vision, information retrieval, and scheduling; see [1,6,9].

Geometric Independent Set. In the geometric setting we are given a set of geometric objects $\mathcal{S}$, and we say a subset $\mathcal{S}' \subseteq \mathcal{S}$ is *independent* if no two objects in $\mathcal{S}'$ intersect and we say $\mathcal{S}'$ is a *clique* if every two objects pairwise intersect. Let $\alpha(\mathcal{S})$ be the cardinality of the largest subset $\mathcal{S}' \subseteq \mathcal{S}$ such that $\mathcal{S}'$ is an independent set and $\omega(\mathcal{S})$ the cardinality of the largest subset of $\mathcal{S}' \subseteq \mathcal{S}$ such that $\mathcal{S}'$ is a clique. The *geometric maximum independent set* and *geometric maximum clique* problem ask for a set $S \subseteq \mathcal{S}$ of independent objects (that induce a clique) such that $S = \alpha(\mathcal{S})$ ($S = \omega(\mathcal{S})$).

Given a set $\mathcal{S}$ of geometric objects, we define the *geometric intersection graph* $\mathcal{G}(S)$ as the simple graph in which each vertex corresponds to an object in $\mathcal{S}$ and two vertices are connected by an edge if their corresponding objects intersect.

Stronger results are known for the geometric maximum independent set problem is known in comparison to general graphs. A fundamental problem is the 1-dimensional case, where all objects are intervals. This problem is also known as *interval selection* problem which has applications to scheduling and resource allocation [7]. For intervals geometric maximum independent set can be solved in $O(n \log n)$ time, by a simple greedy algorithm that sweeps the line from left to right and at each step picks the interval with the leftmost right endpoint, see e.g. [34]. In contrast, the geometric maximum independent set problem is already NP-hard for sets of segments in the plane using only two directions [35], or 2-intervals [8]. For some restricted classes of segment intersection graphs, such as permutation [31] and circle graphs [25], the geometric maximum independent set problem can be solved in polynomial time. Efficient approximation algorithms exist for example for unit square intersection graphs [21] and more generally for pseudo disks [15], as well as segments [3,22]. In a recent breakthrough work [37], it was shown that there exists a constant-factor approximation scheme for maximum independent set for a set of axis-aligned rectangles. Very recently, this factor was improved to 3 [24]. Also, the geometric maximum independent set problem has been extensively studied for dynamic geometric objects, i.e., objects can be inserted and deleted [10,12,17,26,29].

Streaming Algorithms. In this paper, we study the geometric maximum independent set and geometric maximum clique problems for *insertion only streams* of geometric objects. In the streaming model we consider data that is too large to fit at once into the working memory. Instead the data is dealt with in a data stream in which we receive the elements of the input one after another in no specific order and have access to only a limited amount of memory. More specifically, in this model, we have bounds on the amount of available memory. As the data arrives sequentially, and we are not allowed to look at input data of the past, unless the data was stored in our limited memory. This is effectively equivalent to assuming that we can only make one or a few passes over the input

data. We refer to [36,38] and the lecture notes of Chakrabati [13] for an overview on the general topic of streaming algorithms.

For maximum independent set Halldórsson et al. [27] studied the problem for graphs and hypergraps in linear space in the semi-streaming model: Their model work in poly-logarithmic space, like in the case of the classical streaming model, but they can access and update the output buffer, treating it as an extra piece of memory. Kane et al. [32] gave the first optimal algorithm for estimating the number of distinct elements in a data stream.

Streaming algorithms for geometric data have seen a flurry of results in recent years; see [2,16,19,23,30]. Note that one can also view a stream of geometric objects $\mathcal{S}$ as a vertex stream, also called *implicit vertex stream*, of its associated geometric intersection graph $\mathcal{G}(\mathcal{S})$ [18]. Finding an independent, i.e. disjoint, set of geometric objects in a data stream has been among the most studied problems in this geometric direction. Emek et al. [20] studied the interval selection problem, where the input is a set of intervals $\mathcal{I}$ with real endpoints, and the objective is to find an independent subset of largest cardinality. They studied the interval selection problem using $O(\alpha(\mathcal{I}))$ space. They presented a 2-approximation algorithm for the case of arbitrary intervals and a (3/2)-approximation for the case of unit intervals, i.e., when all intervals have the same length. These bounds are also known to be the best possible [20]. Cabello et al. [11] studied the question of estimating $\alpha(\mathcal{I})$ for a set $\mathcal{I}$ of intervals and gave simpler proofs of the algorithms presented by Emek et al. [20].

Cormode et al. [18] considered unit balls in the L^1 and L^∞ norms, i.e. squares in $\mathbb{R}^2$. For a set of such unit balls $\mathcal{B}$ they obtained a 3-approximation using $O(\alpha(\mathcal{B}))$ space and show that there is no $\frac{5}{2} - \varepsilon$ approximation using $o(|\mathcal{B}|)$ space. Finally, Bakshi et al. [5] considered Turnstile streams, i.e., deletion is also allowed, of (weighted) unit intervals and disks.

1.1 Our Results

In this paper we investigate several geometric objects that have not been studied in the context of streaming algorithms. We show in Sect. 2 that there is no constant-factor approximation in the streaming model for finding an independent set of n segments using $o\left(\frac{n}{p}\right)$ bits of memory for any constant number p of passes and this bound holds even if the endpoints of the segments are on two parallel lines. In other words, our bound holds even when the geometric intersection graph of the segments is a permutation graph.

Our construction leads to an interesting consequence. Namely, the intersection graph created in our reduction is not only a permutationa graph, but also an interval graph and the cardinality of its maximum independent set is not dependent on the input size. Since there exists a 2-approximation algorithm for geometric independent set of a set of intervals $\mathcal{I}$ in the streaming model that uses only $O(\alpha(\mathcal{I}))$ space this implies that there is a difference between an interval graph being streamed as a set of intervals or as a set of segments. We discuss this implication in Sect. 3. In Sect. 4 we show that for streams of 2-intervals there is

no one-pass algorithm that achieves a constant-factor approximation using less than $o(n)$ bits. On the positive side we show in Sect. 5 that for n axis-aligned unit height rectangles there exists a one pass streaming algorithm achieving a 4-approximation of the largest set of disjoint rectangles using $O(\alpha(\mathcal{R}) \log n)$ bits.

Finally, we show in Sect. 6 that the distinction between segments and intervals observed for the geometric independent set problem does not occur for the same objects in the geometric clique problem by showing that there does not exist a p-pass algorithm using less than $o\left(\frac{n}{p}\right)$ bits of memory and achieving a constant-factor approximation of the geometric clique problem in streams of n unit intervals. We complement this hardness result by showing how to obtain an exact solution for the geometric clique problem in streams of n intervals using only $O(n \log \omega(\mathcal{I}))$ bits of memory.

2 Independent Sets in Streams of Segments

In this section we establish our lower bound for the memory necessary to approximate the maximum independent set problem to any constant factor on streams of segments. We employ a lower bound reduction technique that uses multi-party set disjointness, which gives us space bounds not only for single-pass algorithms, but also for multi-pass algorithms. The following problem was first studied by Alon, Matias, and Szegedy [4].

Definition 1 (Multi-Party Set Disjointness). *There are t players $P_1, \ldots, P_t$. Each player P_i has a size n bit string x^i. The players want to find out if there is an index $j \in [n]$ where $x^i_j = 1$ for all i. So,* $\textsc{Disj}_{n,t}(x^1, \ldots, x^t) = \bigvee_{j=1}^{n} \bigwedge_{i=1}^{t} x^i_j$.

In our proof we are going to make use of the following result on the communication complexity of Multi-Party Set Disjointness.

Theorem 1 (Chakrabarti et al. [14]). *For an error probability $0 < \delta < 1/4$, to decide $\textsc{Disj}_{n,t}$ the players need $\Omega(\frac{n}{t \log t})$ bits of communication, even for a family of instances $(x^1, \ldots, x^t)$ satisfying the following properties*

$$|\{j : x^i_j = 1\}| = n/2t \qquad \forall i \in [t] \tag{1}$$

$$|\{i : x^i_j = 1\}| \in \{0, 1, t\} \qquad \forall j \in [n] \tag{2}$$

$$|\{j : |\{i : x^i_j = 1\}| = t\}| \leq 1 \tag{3}$$

We can use Theorem 1 by having t players use a streaming algorithm to answer $\textsc{Disj}_{n,t}$. The players construct the stream by creating some part of the stream and giving it to the algorithm, and then passing the memory state of the algorithm to the next player who does the same. This way, the space used by the algorithm must abide to the lower bound on the communication between the players. We can use the t players (rather than just 2) to create a bigger gap between the yes and no answer, excluding the possibility for any constant factor approximation algorithms.

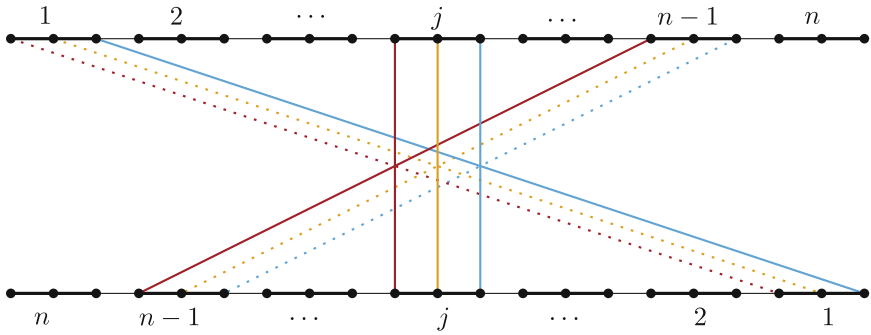

Fig. 1. Lower bound for Independent Set in permutation graphs, with $t = 3$ players. Here the independent set has size t, by the j-th group, where $x_j^i = 1$ for all $i \in [t]$.

Lemma 1. *For any $p \geq 1, t \geq 2$, any algorithm for geometric maximum independent set that can distinguish between instances with an independent set of size 1 and t and succeeds with probability at least $3/4$ on segment streams using p passes must use at least $\Omega(\frac{n}{p \cdot t \log t})$ bits of memory, even when the segment endpoints lie on two parallel lines.*

Proof. Let $(x^1, \ldots, x^t)$ be an instance of DISJOINTNESS with t players, each with n bits. We construct a permutation graph depending on the input to DISJOINTNESS, as illustrated in Fig. 1 for $t = 3$. Let the permutation graph have n groups of t points both on the top, labelled $1, \ldots, n$ from left to right. On the bottom do the same, but we label from n to 1 from left to right. For $i \in [t], j \in [n]$, player i creates a segment from the i-th point in group j at the top to the i-th point in group j at the bottom when $x_j^i = 1$. This creates a permutation graph with $n' = n/2$ vertices by Property 1 of Theorem 1.

The players construct the segment stream for some algorithm for MAX-CLIQUE as follows, starting from player 1, player i inputs all their $n/2t$ segments, then passes the memory state of the algorithm to player $i+1$, and this continues until all players have input their segments.

We claim that the graph will contain an independent set of size t if exactly $\text{DISJ}_{n,t}(x^1, \ldots, x^t) = 1$, and otherwise the maximum independent set size is 1.

First notice that any segment inserted to a group $j \in [n]$ intersects all other segments in the graph, except for segments inserted to group j, as these segments are parallel to it. Hence, any independent set can only contain vertices that correspond to segments inserted to the same group j, for some $j \in [n]$, and the size of the independent set is the number of 1's present over all players at index j. Now by Property 2 of Theorem 1, any independent set can have only size 1 or t in the graph. And indeed, an independent set of size t implies that all t players inserted a segment for some group $j \in [n]$, and hence all have a 1 for index j.

Now it follows from Theorem 1 that any algorithm for maximum independent set on a permutation graph with n' vertices that can discern between indepen-

dent set size 1 and t with probability at least $3/4$ using p passes over the stream must use at least $\Omega(\frac{n}{p \cdot t \log t}) = \Omega(\frac{n'}{p \cdot t \log t})$ bits of memory. □

We now use Lemma 1 to give a general hardness statement for approximation geometric maximum independent set in segment streams.

Theorem 2. *Any constant-factor approximation algorithm for geometric maximum independent set that succeeds with probability at least* $3/4$ *on segment streams using* p *passes must use at least* $\Omega(n/p)$ *bits of memory, even when it is known that the segments correspond to a permutation graph.*

Proof. Let us be given some constant-factor approximation algorithm for geometric maximum independent set that succeeds with probability at least $3/4$ on segment streams of permutation graphs in p passes. Then there exists some $c \geq 2$ such that the algorithm can distinguish between an independent set of size 1 or c in a given graph. But then we can apply Lemma 1 to get that this algorithm must use at least $\Omega(\frac{n}{p \cdot c \log c}) = \Omega(n/p)$ bits of memory. □

3 Intervals and Segments are Different

When we consider the intersection graph G of the set of segments constructed in the proof of Theorem 2 one can see that it has a straight-forward representation as a set of intervals whose intersection graph is isomorphic to G. Also, notice that the size of the independent set of the construction is not dependent on the length of the bit strings, but only on the number of players. For example, we can rule out the existence of a 2-approximation streaming algorithm using any constant number p of passes and $o(n/p)$ bits of memory, already when $t/2 \geq 2 \iff t \geq 4$ players are used in the construction presented in Lemma 1.

At the same time there exists a 2-approximation one pass streaming algorithm for independent sets of intervals that uses only $O(\alpha(\mathcal{I}) \log |\mathcal{I}|)$ bits of memory where $\mathcal{I}$ is the set of input intervals [11,20]. Hence, these algorithms find a 2-approximation of the independent set of the intersection graph constructed in the proof of Lemma 1 using only $O(\log n)$ bits of memory if the graph was given as a set of intervals. This leads to the following corollary.

Corollary 1. *Given a stream of segments* $\mathcal{S}$ *whose intersection graph* $\mathcal{G}(\mathcal{S})$ *is in the intersection of permutation and interval graphs, there is no algorithm that uses* $o(n/p)$ *bits of memory and* $p \geq 1$ *passes and computes a stream of intervals* $\mathcal{I}$ *such that* $\mathcal{G}(\mathcal{I})$ *is isomorphic to* $\mathcal{G}(\mathcal{S})$.

4 Independent Sets in Streams of c-Intervals

A *c-interval* is a set of non-overlapping intervals $\{I_1, \ldots, I_c\}$ on the real line. We say two c-intervals intersect if at least two of their intervals have one point in common. We call a family of c-intervals *separated* if the intervals can be split into independent groups of intervals, each containing at most one interval from each

c-interval, without changing which c-intervals intersect. Since we only consider 2-intervals we can talk of left and right intervals. Formally, let $T = \{L, R\}$ be a 2-interval such that the startpoint of L is left of the startpoint of R, then we denote L as the *left* interval of T and R as the *right* interval of T.

For the reduction we use the $\textsc{Chain}_t$ communication problem which was introduced by Cormode et al. [18].

Definition 2 (Cormode et al. [18]). *The t-party chained index problem $\textsc{Chain}_t$ consists of $t-1$ n-bit binary vectors $\{x^i\}_{i=1}^{t-1}$, along with corresponding indices $\{\sigma_i\}_{i=1}^{t-1}$ from the range $[n]$. We have the promise that the entries $\{x^i_{\sigma_i}\}_{i=1}^{t-1}$ are all equal to the desired bit $z \in \{0,1\}$. The input is initially allocated as follows:*

- *The first party P_1 knows x^1*
- *Each intermediate party P_p for $1 < p < k$ knows x^p and σ_{p-1}*
- *The final party P_t knows just σ_{t-1}*

Communication proceeds as follows: P_1 sends a single message to P_2, then P_2 communicates to P_3, and so on, with each party sending exactly one message to its immediate successor. After all messages are sent, P_k must correctly output z, succeeding with probability at least $2/3$. If the promise condition is violated, any output is considered correct.

Cormode et al. [18] showed the following result in the same paper.

Theorem 3 (Cormode et al. [18]). *Any communication scheme $\mathcal{B}$ which solves $\textsc{Chain}_t$ must communicate at least $\Omega(\frac{n}{t^2})$ bits.*

In the following we use the $\textsc{Chain}_t$ to show that there is no one-pass streaming algorithm that computes a constant-factor approximation of the maximum independent set for a family of separated 2-intervals using $o(n)$ bits of memory.

Remark 1. For $\textsc{Chain}_t$ we may assume that party $i > 1$ knows all indices before σ_{i-1}. To realize this, just assume that every party i appends all $i-1$ previous indices to its message. This uses only $O(t \log n)$ bits in each such message and hence $O(t^2 \log n)$ bits in total. As this is a lower order term with respect to the bound in Theorem 3 we retain the linear communication bound of $\Omega(\frac{n}{t^2})$.

We define an *interval stack* as a set of intervals $I_1, \ldots, I_n$ on the real line where first all startpoints appear in order of the indices and then all endpoints, again in order of the interval indices. We denote as *left-gap* the space between the startpoint of I_i and I_{i+1} for $i = 1, \ldots, n-1$ and the startpoint of I_n and the endpoint of I_1. Observe that any interval containing a point of the left gap of I_i intersects all intervals I_j with $1 \leq j \leq i$. As for independent sets of segments we first show a technical lemma.

Lemma 2. *For any $t \geq 2$, any algorithm for geometric maximum independent set that can distinguish between instances admitting an independent set of size 1 and t and succeeds with probability at least $2/3$ on streams of 2-intervals must use at least $\Omega(\frac{n}{t^3})$ bits of memory, even if the union of the 2-intervals is separated.*

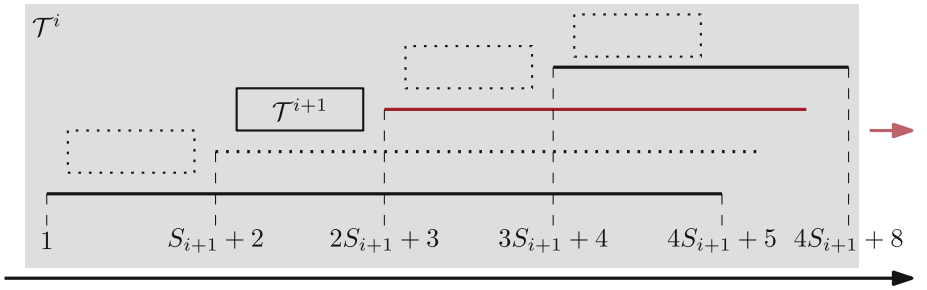

Fig. 2. Left intervals created for party i in the construction of Lemma 2. The intervals in the grey box are for party i, the intervals outside are of party $i-1$, and the boxes mark the left gap spaces for party $i+1$. The red interval is the interval corresponding to a 1 bit at σ_i. Dotted intervals and boxes are not actually inserted by the parties. The shown coordinates are the local coordinates for the interval stack inserted by party i. (Color figure online)

Proof. Given an instance of $\textsc{Chain}_t$ with $\frac{n}{t}$-length bit strings x^i and indices σ_j. Let $N = \frac{n}{t}$ and will assume for simplicity that $\frac{n}{t}$ is a whole number. We create one 2-interval $T^i_j = (L^i_j, R^i_j)$ for each 1 bit at index j of bit string i and one additional 2-interval for party t. In the following we first describe the construction of the left intervals. See Fig. 2 for an illustration.

Create an interval stack $\mathcal{L}^i = \{L^i_j \mid \forall j \in [N] : x^i \text{ has a 1 bit at index } j\}$. To simplify the presentation we assume that all $\frac{n}{t}$ intervals are present in $\mathcal{T}^i$. When actually constructing the intervals in a stream party i simply does not add an interval when the jth bit is set to 0, but still shifts the coordinates accordingly. We initially place the intervals of $\mathcal{L}^1$ and then place $\mathcal{L}^i$ for $i > 1$ in the left gap of $L^{i-1}_{\sigma_{i-1}}$. For player t we add one interval L^t_1 in the left gap of $L^{t-1}_{\sigma_{t-1}}$. Let $\mathcal{L}$ be the union of all $\mathcal{L}^i$.

To complete the construction we create the same construction using the reversed bit strings for each party. This creates the interval stacks $\mathcal{R}^i, i = 1, \ldots, n$ and with R^i_j we denote the right interval inserted by the ith party for the 1 bit at index j in the non-reversed bit string x^i. Let $\mathcal{R}$ be the union of all $\mathcal{R}^i$. Finally, we create a set of 2-intervals as $\mathcal{T} = \{(L^i_j, R^i_j) \mid L^i_j \in \mathcal{L}^i \text{ and } R^i_j \in \mathcal{R}^i\}$.

Consider a 2-interval $T^i_j = (L, R)$ inserted for bit string x^i such that $j \neq \sigma_i$ for any $i \in \{1, \ldots, t-1\}$. Then, L is contained in all intervals L^a_b with $a < i$ and $b < \sigma_a$. Moreover, L contains every L^a_b with $a > i$ and $b < \sigma_i$. Similarly, R is contained in all intervals R^a_b with $a > i$ and $b > \sigma_j$ and contains every R^a_b with $a > i$ and $b > \sigma_i$. Consequently if T^i_j is part of an independent set $\mathcal{I} \subseteq \mathcal{T}$ we can only add 2-intervals T^a_b to $\mathcal{I}$ with $a < i$ and $j = \sigma_a$.

If the answer bit is 0 then no 2-interval corresponding to some σ_i index exists and hence by the above argumentation the largest independent set has size one. If the answer bit is 1 then there is a 2-interval $T^i_{\sigma_i}$ for every σ_i with $i = 1, \ldots, t$ and all t of them are independent. Hence, the largest independent set in this case has size t.

It remains to describe the precise coordinates of the intervals and argue that we only need $O(\log n)$ bits to represent the construction for every fixed t. For each party $i = 1, \ldots, t$ let S_i be the length of the left interval stack. Since for party t we insert only one 2-interval we set $S_t = 2$. Let $\mathcal{T}$ be the set of 2-intervals created as above. In the following we consider only the left intervals of every $T_j^i \in \mathcal{T}$. The calculation and placement works analogously for the right intervals after reversing every x^i. Let L_j^i be a left interval for party i and index j. We put the startpoint of L_j^i at position $1+(j-1)\cdot(S_{i+1}+1)$ and its endpoint at $j + N \cdot (S_{i+1}+1)$. Hence, for party i its left interval stack has length at most

$$S_i = N + N \cdot (S_{i+1}+1) = N \cdot (S_{i+1}+2).$$

This can be written as a closed formula

$$S_i = 4N^{t-i} \cdot 2 \sum_{j=1}^{t-(i+1)} N^j = 4N^{t-i} \cdot 2 \left(\frac{N^{t-i} - N}{N-1} \right).$$

Now, party i places its left stack at

$$P_i = 1 + \sum_{j=1}^{i-1} ((\sigma_j - 1) \cdot (S_{j+1}+1) + 1).$$

The last party places an interval of length two at position P_t. Since every left interval placed by a party $i > 1$ is nested by the intervals inserted into the stream by the first party we can conclude that $S_1 \in O\left(N^{t-1}\right)$. This number can be represented using $O(t \log N) = O(t \log \frac{n}{t})$ bits. Since t can be treated as a constant we get that we only require $O(t \log \frac{n}{t}) = O(\log n)$ bits. □

We conclude Theorem 4 from Lemma 2 in the same way as for Theorem 2.

Theorem 4. *Any constant-factor approximation algorithm for geometric maximum independent set that succeeds with probability at least* 2/3 *on streams of* 2*-intervals requires at least* $\Omega(n)$ *bits of memory, even if the* 2*-intervals are separated.*

5 Independent Sets in Streams of Unit-Height Rectangles

In this section, we study the independent set problem for a stream of unit height arbitrary width rectangles. To conform with previous work we assume in this section that one cell of memory can store one rectangle, i.e., one cell of memory has $\Theta(\log n)$ bits where all coordinates of the rectangles are assumed to be in $O(n)$. Cabello and Pérez-Lantero [11] studied the independent set problem for streams of intervals on the real line and achieved the following result.

Theorem 5 (Theorem 5 [11]). *Let $\mathcal{I}$ be a set of intervals in the real line that arrive in a data stream. There is a data stream algorithm to compute a* 2*-approximation to the largest independent subset of $\mathcal{I}$ that uses $O(\alpha(\mathcal{I}))$ space and handles each interval of the stream in $O(\log \alpha(\mathcal{I}))$ time.*

Using Theorem 5 we obtain a constant-factor approximation for finding the largest independent set of rectangles in a stream of axis-aligned unit height rectangles in one pass using $O(\alpha(\mathcal{R}))$ space. The below notation is similar to the one used by Cabello and Pérez-Lantero [11].

We divide the y-axis into size two intervals. Similar to [11] we define windows $W_\ell = [\ell, \ell + 2)$. Then, we form two partitions $\mathcal{W}_0$ and $\mathcal{W}_1$ of the y-axis as $\mathcal{W}_z = \{W_{z+2i} \mid i \in \mathbb{Z}\}$ for $z \in \{0, 1\}$. We denote with $\mathcal{R}_z \subseteq \mathcal{R}$ and $z \in \{0, 1\}$ the set of rectangles that is contained in any window of $\mathcal{W}_z$. Observe, that every rectangle is fully contained in only one of the two partitions.

Computing an independent set for the rectangles $\mathcal{R}_z$ now amasses to computing independent sets for each set of rectangles lying in one window w_ℓ of W_z. By only considering windows that contain at least one interval and using Theorem 5 we can compute for every W_z and $z \in \{0, 1\}$ a 2-approximation of its largest independent set using $\alpha(\mathcal{R}_z)$ space in one pass. Let $\alpha'(\mathcal{R}_z)$ be such a 2-approximation, $\alpha(\mathcal{R}_z)$ the size of an optimal independent set of $\mathcal{R}_z$, and $\mathcal{R}_I \subseteq \mathcal{R}$ an optimal independent set of $\mathcal{R}$, then it holds that

$$\begin{aligned} 2\max\{\alpha'(\mathcal{R}_0), \alpha'(\mathcal{R}_1)\} &\geq \alpha'(\mathcal{R}_0) + \alpha'(\mathcal{R}_1) \geq \frac{1}{2}(\alpha(\mathcal{R}_0) + \alpha(\mathcal{R}_1)) \\ &\geq \frac{1}{2}(|\mathcal{R}_I \cap \mathcal{R}_0| + |\mathcal{R}_I \cap \mathcal{R}_1|) \geq \frac{1}{2}|\mathcal{R}_I| \geq \frac{1}{2}\alpha(\mathcal{R}). \end{aligned}$$

From this it follows that $\max\{\alpha'(\mathcal{R}_0), \alpha'(\mathcal{R}_1)\} \geq \frac{1}{4}\alpha(\mathcal{R})$.

Theorem 6. *Let $\mathcal{R}$ be a set of axis-aligned unit height rectangles that arrive in a data stream, there is an algorithm that compute a 4-approximation to the maximum independent set of $\mathcal{R}$, uses $O(\alpha(\mathcal{R}))$ space, and handles each rectangle in polylog time.*

Note, that this algorithm restricted to axis-aligned squares matches the approximation factor of three due to Cormode et al. [18] since for unit intervals we can use the $\frac{3}{2}$-approximation algorithm from Cabello et al. [11].

6 Clique in Streams of Intervals and Segments

We can make an identical statement as Theorem 2 for maximum clique instead of maximum independent set by observing the complement graph of the construction in Lemma 1.

Theorem 7. *Any constant-factor approximation algorithm for geometric maximum clique that succeeds with probability at least $3/4$ on segment streams using p passes must use at least $\Omega(n/p)$ bits of memory, even when the endpoints of the segments lie on two lines.*

Proof. The complement graph of the construction of Lemma 1 is also a permutation graph and admits the property that it contains either a clique of size 1 or t. It is given by reversing the permutation (exactly mirroring the bottom) of the

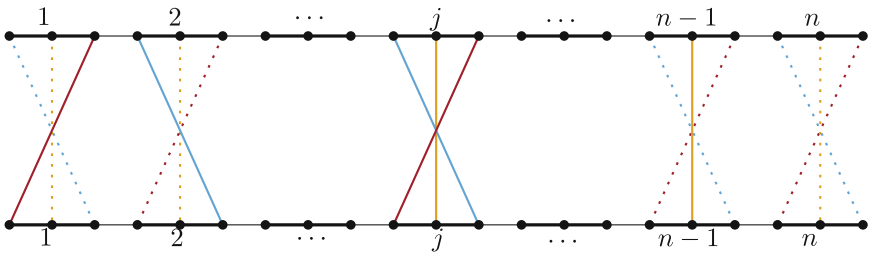

Fig. 3. Lower bound for Clique in permutation graphs, with $t = 3$ players. In this example, $x_j^i = 1$ for all players $i \in [t]$.

construction in Lemma 1. This construction is illustrated in Fig. 3. It follows that Lemma 1 also holds for geometric maximum clique instead of geometric maximum independent set. The theorem now follows from the proof of Theorem 2, using maximum clique instead of maximum independent set. □

For streams of intervals, we show a simple upper bound, using that there are at most $2n$ different endpoints of intervals.

Theorem 8. *Let $\mathcal{I}$ be a set of intervals in the real line that arrive in a data stream. There is an algorithm to compute the largest clique size, $\omega(\mathcal{I})$, in 1 pass using $O(n \log(\omega(\mathcal{I}))$ bits of memory, using time $O(n^2)$ total. In a second pass, the intervals that make up the clique can be recovered, which can be streamed without extra memory use, or stored using $O(\omega(\mathcal{I}) \log n)$ bits of memory.*

Proof. We keep a counter for every possible endpoint of an interval, which are $2n$ counters total. We keep the order of counters fixed, but need no labels for a counter, because of the assumption that the range of endpoints is $1, \ldots, 2n$. When an interval appears in the stream, we increment all counters that are contained in the interval, including its endpoints. At the end of the stream, $\omega(\mathcal{I})$ is given by the largest counter, as this coordinate is a witness to $\omega(\mathcal{I})$ intervals co-intersecting. This is the correct maximum, as the number of intersecting intervals can only change at an endpoint of an interval.

In the second pass, we can recover the intervals that make up the clique can be recovered by pushing every interval that overlaps the coordinate of the maximum counter found in the first pass to the output. □

The result of Theorem 8 is nearly tight, as the construction of Theorem 7 can also be constructed as a stream of unit intervals.

7 Conclusion

We studied the geometric independent set and clique problems for a variety of geometric objects. Interestingly, we showed that the type of geometric object

used for the implicit stream of a geometric intersection graph can make a substantial difference even for simple objects like segments and intervals. This raises the question if such a difference also exists for other types of objects. Moreover, the complexity of finding an independent set in a stream of arbitrary rectangles remains open. Finally, studying streams of geometric objects in other streaming models, such as turnstile streams, provides an interesting direction for future research.

References

1. Agarwal, P.K., van Kreveld, M.J., Suri, S.: Label placement by maximum independent set in rectangles. Comput. Geom.: Theory Appl. **11**(3–4), 209–218 (1998). https://doi.org/10.1016/S0925-7721(98)00028-5
2. Agarwal, P.K., Krishnan, S., Mustafa, N.H., Venkatasubramanian, S.: Streaming geometric optimization using graphics hardware. In: Di Battista, G., Zwick, U. (eds.) ESA 2003. LNCS, vol. 2832, pp. 544–555. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39658-1_50
3. Agarwal, P.K., Mustafa, N.H.: Independent set of intersection graphs of convex objects in 2D. Comput. Geom.: Theory Appl. **34**(2), 83–95 (2006). https://doi.org/10.1016/j.comgeo.2005.12.001
4. Alon, N., Matias, Y., Szegedy, M.: The space complexity of approximating the frequency moments. J. Comput. Syst. Sci. **58**(1), 137–147 (1999). https://doi.org/10.1006/jcss.1997.1545
5. Bakshi, A., Chepurko, N., Woodruff, D.P.: Weighted maximum independent set of geometric objects in turnstile streams. In: Proceedings of the Annual International Conference on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020). LIPIcs, vol. 176, pp. 64:1–64:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPIcs.APPROX/RANDOM.2020.64
6. Balas, E., Yu, C.S.: Finding a maximum clique in an arbitrary graph. SIAM J. Comput. **15**(4), 1054–1068 (1986). https://doi.org/10.1137/0215075
7. Bar-Noy, A., Bar-Yehuda, R., Freund, A., Naor, J., Schieber, B.: A unified approach to approximating resource allocation and scheduling. J. ACM **48**(5), 1069–1090 (2001). https://doi.org/10.1145/502102.502107
8. Bar-Yehuda, R., Halldórsson, M.M., Naor, J., Shachnai, H., Shapira, I.: Scheduling split intervals. SIAM J. Comput. **36**(1), 1–15 (2006). https://doi.org/10.1137/S0097539703437843
9. van Bevern, R., Mnich, M., Niedermeier, R., Weller, M.: Interval scheduling and colorful independent sets. J. Sched. **18**(5), 449–469 (2014). https://doi.org/10.1007/s10951-014-0398-5
10. Bhore, S., Cardinal, J., Iacono, J., Koumoutsos, G.: Dynamic geometric independent set. CoRR abs/2007.08643 (2020). https://arxiv.org/abs/2007.08643
11. Cabello, S., Pérez-Lantero, P.: Interval selection in the streaming model. Theor. Comput. Sci. **702**, 77–96 (2017). https://doi.org/10.1016/j.tcs.2017.08.015
12. Cardinal, J., Iacono, J., Koumoutsos, G.: Worst-case efficient dynamic geometric independent set. In: Proceedings of the 29th Annual European Symposium on Algorithms (ESA 2021). LIPIcs, vol. 204, pp. 25:1–25:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPIcs.ESA.2021.25

13. Chakrabarti, A.: CS49: data stream algorithms lecture notes (2020). http://www.cs.dartmouth.edu/ac/Teach/data-streams-lecnotes.pdf
14. Chakrabarti, A., Khot, S., Sun, X.: Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In: 18th Annual IEEE Conference on Computational Complexity (Complexity 2003), Aarhus, Denmark, 7–10 July 2003, pp. 107–117. IEEE Computer Society (2003). https://doi.org/10.1109/CCC.2003.1214414
15. Chan, T.M., Har-Peled, S.: Approximation algorithms for maximum independent set of pseudo-disks. Discret. Comput. Geom. **48**(2), 373–392 (2012). https://doi.org/10.1007/s00454-012-9417-5
16. Chen, X., Jayaram, R., Levi, A., Waingarten, E.: New streaming algorithms for high dimensional EMD and MST. In: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2022), pp. 222–233. ACM (2022). https://doi.org/10.1145/3519935.3519979
17. Compton, S., Mitrovic, S., Rubinfeld, R.: New partitioning techniques and faster algorithms for approximate interval scheduling. CoRR abs/2012.15002 (2020). https://arxiv.org/abs/2012.15002
18. Cormode, G., Dark, J., Konrad, C.: Independent sets in vertex-arrival streams. In: Proceedings of the 46th International Colloquium on Automata, Languages, and Programming, (ICALP 2019). LIPIcs, vol. 132, pp. 45:1–45:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019). https://doi.org/10.4230/LIPIcs.ICALP.2019.45
19. Czumaj, A., Jiang, S.H., Krauthgamer, R., Veselý, P.: Streaming algorithms for geometric steiner forest. CoRR abs/2011.04324 (2020). https://arxiv.org/abs/2011.04324
20. Emek, Y., Halldórsson, M.M., Rosén, A.: Space-constrained interval selection. ACM Trans. Algorithms **12**(4), 51:1–51:32 (2016). https://doi.org/10.1145/2886102
21. Erlebach, T., Jansen, K., Seidel, E.: Polynomial-time approximation schemes for geometric intersection graphs. SIAM J. Comput. **34**(6), 1302–1323 (2005). https://doi.org/10.1137/S0097539702402676
22. Fox, J., Pach, J.: Computing the independence number of intersection graphs. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2011), pp. 1161–1165. SIAM (2011)
23. Frahling, G., Sohler, C.: Coresets in dynamic geometric data streams. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC 2005), pp. 209–217. ACM (2005). https://doi.org/10.1145/1060590.1060622
24. Gálvez, W., Khan, A., Mari, M., Mömke, T., Pittu, M.R., Wiese, A.: A 3-approximation algorithm for maximum independent set of rectangles. In: Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, (SODA 2022), pp. 894–905. SIAM (2022). https://doi.org/10.1137/1.9781611977073.38
25. Gavril, F.: Algorithms for a maximum clique and a maximum independent set of a circle graph. Networks **3**(3), 261–273 (1973). https://doi.org/10.1002/net.3230030305
26. Gavruskin, A., Khoussainov, B., Kokho, M., Liu, J.: Dynamic algorithms for monotonic interval scheduling problem. Theor. Comput. Sci. **562**, 227–242 (2015). https://doi.org/10.1016/j.tcs.2014.09.046
27. Halldórsson, B.V., Halldórsson, M.M., Losievskaja, E., Szegedy, M.: Streaming algorithms for independent sets. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6198, pp. 641–652. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14165-2_54

28. Håstad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. In: Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS 1996), pp. 627–636. IEEE Computer Society (1996). https://doi.org/10.1109/SFCS.1996.548522
29. Henzinger, M., Neumann, S., Wiese, A.: Dynamic approximate maximum independent set of intervals, hypercubes and hyperrectangles. In: Proceedings of the 36th International Symposium on Computational Geometry (SoCG 2020). LIPIcs, vol. 164, pp. 51:1–51:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPIcs.SoCG.2020.51
30. Indyk, P.: Streaming algorithms for geometric problems. In: Lodaya, K., Mahajan, M. (eds.) FSTTCS 2004. LNCS, vol. 3328, pp. 32–34. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30538-5_3
31. Trotter, L.E., Jr.: Algorithmic graph theory and perfect graphs, by Martin C. Golumbic, academic, New York, 284 pp. price: $34.00. Networks **13**(2), 304–305 (1983). https://doi.org/10.1002/net.3230130214
32. Kane, D.M., Nelson, J., Woodruff, D.P.: An optimal algorithm for the distinct elements problem. In: Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2010), pp. 41–52. ACM (2010). https://doi.org/10.1145/1807085.1807094
33. Karp, R.M.: Reducibility among combinatorial problems. In: Proceedings of a Symposium on the Complexity of Computer Computations. The IBM Research Symposia Series, pp. 85–103. Plenum Press, New York (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
34. Kleinberg, J.M., Tardos, É.: Algorithm Design. Addison-Wesley, Boston (2006)
35. Kratochvíl, J., Nešetřil, J.: Independent set and clique problems in intersection-defined classes of graphs. Comment. Math. Univ. Carol. **31**(1), 85–93 (1990)
36. McGregor, A.: Graph stream algorithms: a survey. SIGMOD Rec. **43**(1), 9–20 (2014). https://doi.org/10.1145/2627692.2627694
37. Mitchell, J.S.B.: Approximating maximum independent set for rectangles in the plane. In: Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, pp. 339–350. IEEE (2021). https://doi.org/10.1109/FOCS52979.2021.00042
38. Muthukrishnan, S., et al.: Data streams: algorithms and applications. Found. Trends® Theor. Comput. Sci. **1**(2), 117–236 (2005)
39. Zuckerman, D.: Linear degree extractors and the inapproximability of max clique and chromatic number. Theory Comput. **3**(1), 103–128 (2007). https://doi.org/10.4086/toc.2007.v003a006