Random-Order Online Independent Set of Intervals and Hyperrectangles

Mohit Garg
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Debajyoti Kar
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Arindam Khan
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Abstract

In the Maximum Independent Set of Hyperrectangles problem, we are given a set of n (possibly overlapping) d-dimensional axis-aligned hyperrectangles, and the goal is to find a subset of non-overlapping hyperrectangles of maximum cardinality. For $d = 1$, this corresponds to the classical Interval Scheduling problem, where a simple greedy algorithm returns an optimal solution. In the offline setting, for d-dimensional hyperrectangles, polynomial time $(\log n)^{O(d)}$-approximation algorithms are known [16]. However, the problem becomes notably challenging in the online setting, where the input objects (hyperrectangles) appear one by one in an adversarial order, and on the arrival of an object, the algorithm needs to make an immediate and irrevocable decision whether or not to select the object while maintaining the feasibility. Even for interval scheduling, an $\Omega(n)$ lower bound is known on the competitive ratio.

To circumvent these negative results, in this work, we study the online maximum independent set of axis-aligned hyperrectangles in the random-order arrival model, where the adversary specifies the set of input objects which then arrive in a uniformly random order. Starting from the prototypical secretary problem, the random-order model has received significant attention to study algorithms beyond the worst-case competitive analysis (see the survey by Gupta and Singla [40]). Surprisingly, we show that the problem in the random-order model almost matches the best-known offline approximation guarantees, up to polylogarithmic factors. In particular, we give a simple $(\log n)^{O(d)}$-competitive algorithm for d-dimensional hyperrectangles in this model, which runs in $\tilde{O}_d(n)$ time. Our approach also yields $(\log n)^{O(d)}$-competitive algorithms in the random-order model for more general objects such as d-dimensional fat objects and ellipsoids. Furthermore, all our competitiveness guarantees hold with high probability, and not just in expectation.

2012 ACM Subject Classification Theory of computation → Online algorithms; Theory of computation → Computational geometry; Theory of computation → Scheduling algorithms

Keywords and phrases Online Algorithms, Random-Order Model, Maximum Independent Set of Rectangles, Hyperrectangles, Fat Objects, Interval Scheduling

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.58

Related Version *Full Version*: `https://arxiv.org/abs/2402.14201` [37]

Funding *Mohit Garg*: Supported by a fellowship from the Walmart Center for Tech Excellence at IISc (CSR Grant WMGT-23-0001).
Debajyoti Kar: Supported by a Google PhD Fellowship.
Arindam Khan: Supported in part by Google India Research Award, SERB Core Research Grant (CRG/2022/001176) on "Optimization under Intractability and Uncertainty", and the Walmart Center for Tech Excellence at IISc (CSR Grant WMGT-23-0001).

Acknowledgements The authors wish to thank Jaikumar Radhakrishnan for generously sharing his insights on hypergeometric concentration bounds, Rahul Saladi for helping in constructing fast and elegant data structures, and K.V.N. Sreenivas for some helpful discussions.

1 Introduction

Geometric intersection graphs are extensively studied in discrete geometry, graph theory, and theoretical computer science. In the intersection graph of a collection of geometric objects, the objects form the vertices, and two vertices are connected by an edge if and only if the corresponding objects intersect. The maximum independent set problem for geometric intersection graphs is a fundamental problem in computational geometry. In an equivalent geometric formulation of the problem, given a set $\mathcal{H}$ of n possibly overlapping geometric objects in d-dimensional Euclidean space, our goal is to select a subset $\mathcal{H}^* \subseteq \mathcal{H}$ of maximum cardinality such that the objects in $\mathcal{H}^*$ are pairwise disjoint. Apart from its practical applications in data mining [30], map labeling [23], routing [54], etc., studying the problem has led to the development of several techniques in geometric approximation algorithms [43].

For interval graphs (i.e., when the objects are intervals), the problem is also known as interval scheduling, where a simple greedy algorithm is optimal. In contrast, the problem is W[1]-hard (with respect to the cardinality of the optimal solution) even for intersection graphs of unit disks and axis-aligned unit squares in the plane [56], making the existence of an efficient polynomial-time approximation scheme (EPTAS) unlikely. However, polynomial-time approximation schemes (PTAS) have been established for fat objects, including squares and disks [25, 17]. In the case of d-dimensional axis-aligned hyperrectangles ($d \geq 3$), the current best approximation ratio stands at $O((\log n)^{d-2} \cdot \log\log n)$ [16]. For axis-aligned rectangles, Mitchell [58] provided the first $O(1)$-approximation algorithm, and subsequently, a $(2+\varepsilon)$-approximation was achieved [35, 36]. However, addressing the problem for arbitrary line segments, not necessarily axis-parallel, remains significantly challenging, with only a polynomial-time n^{ε}-approximation currently available [28, 19].

These problems have also recently received attention in the online setting, where the input is revealed to the algorithm by an adversary in parts; upon receiving a part, the algorithm must make an irrevocable decision for that part while adhering to the constraints of the problem, before any other parts are revealed. The adversary stops revealing the input at a certain step, at which point the performance of the algorithm can be measured in terms of its *competitive ratio* – the ratio between the value obtained by the algorithm and the optimum offline value possible for that input. Any online algorithm, even if randomized, for interval scheduling faces an $\Omega(n)$ lower bound on its competitive ratio in this adversarial model [42].

Random-order models (ROM) were introduced to mitigate excessively pessimistic results in the adversarial-order arrival model, presenting a more realistic representation in many scenarios. In this model, the input set of items is chosen by the adversary; however, the arrival order of the items is determined by a permutation selected uniformly at random from the set of all permutations. This random reshuffling of the input items often weakens the adversary, resulting in improved performance guarantees. The competitive ratio of an online algorithm in the random-order arrival model, also known as the *random-order ratio*, is the ratio between the expected value obtained by the algorithm and the optimum offline value. Moreover, it is often preferable to have a high probability guarantee on the value obtained by the algorithm rather than only a guarantee on its expectation. ROM encompasses various other commonly used stochastic models, such as the IID model (where the input sequence consists of independent and identically distributed random samples drawn from some fixed but unknown distribution) and is closely related to the prophet model. An algorithm in ROM also implies an algorithm under the IID model with, at most, the same competitive ratio. For an in-depth discussion, see the survey on these models by Gupta and Singla [40]. Many problems related to scheduling [3, 4, 2, 6, 5, 38, 44] have received recent attention in this model.

Formally, an online algorithm for the maximum independent set problem in the random-order model receives an input set $\mathcal{T}$ consisting of d-dimensional geometric objects in multiple steps. Initially, the algorithm is provided with the total number of objects, $n = |\mathcal{T}|$.[1] Subsequently, the objects from $\mathcal{T}$ are presented to the algorithm in a uniformly random order, one at a time. Upon receiving an object from $\mathcal{T}$, the algorithm must make an irrevocable decision on whether or not to include the given object in the solution before receiving any further objects. The algorithm's objective is to maximize the size of the solution subject to the constraint that the selected objects are pairwise disjoint. We say an algorithm, ALG, is c-competitive ($c > 1$) in the random-order model if, for all inputs $\mathcal{T}$, $\mathbb{E}[|\mathsf{ALG}(\mathcal{T})|] \geq \frac{1}{c} \cdot \mathsf{opt}(\mathcal{T}) - k$. Here, $\mathsf{opt}(\mathcal{T})$ represents the size of a maximum cardinality subset of $\mathcal{T}$ consisting of disjoint objects, and k is some constant independent of $|\mathcal{T}|$. The expectation is taken over the randomness in the arrival order and any coin tosses made by the algorithm. We say an algorithm, ALG, is *strictly* c-competitive ($c > 1$) in the random-order model (also called the absolute random-order ratio in [5]) if, for all inputs $\mathcal{T}$, $\mathbb{E}[|\mathsf{ALG}(\mathcal{T})|] \geq \frac{1}{c} \cdot \mathsf{opt}(\mathcal{T})$. In contrast to offline randomized algorithms, where repeated runs amplify the success probability, in the online setting, expectation guarantees do not readily translate into high-probability guarantees. Such high probability guarantees are often needed and thus studied for online algorithms (e.g., [53, 52, 10, 57]). In this work, the study of such a stronger notion in the context of the random-order arrival model is initiated. We say that ALG is *strongly* c-competitive if for all inputs $\mathcal{T}$, $\mathsf{Pr}[|\mathsf{ALG}(\mathcal{T})| \geq \frac{1}{c} \cdot \mathsf{opt}(\mathcal{T})] = 1 - o(1)$, where the $o(1)$ is with respect to $|\mathcal{T}|$. Under this stronger notion of competitiveness, it is evident that numerous classical problems and algorithms remain to be explored.

However, even in ROM, obtaining an $n^{o(1)}$-competitive algorithm for interval scheduling is nontrivial. In fact, the trivial greedy algorithm has a competitive ratio $n^{\Omega(1)}$ (see the full version [37] for a proof). Göbel et al. [38] studied the problem on graphs with bounded inductive independence number[2] ρ, and designed an online algorithm with a competitive ratio $O(\rho^2)$, *in expectation.* This already implies $O(1)$-competitiveness for interval scheduling. However, they did not provide any high-probability guarantee. In fact, we show in the full version [37] that no online algorithm for interval scheduling that has an initial *observation phase* where it does not pick any of the initial cn intervals (for some absolute constant $c \in (0, 1)$) can be strongly $O(1)$-competitive. In particular, their algorithm is not strongly $O(1)$-competitive for interval scheduling. Additionally, their result does not yield a competitive algorithm for rectangles, where the inductive independence number can be $\Omega(n)$.

Recently, Henzinger et al. [45] studied fully dynamic approximation algorithms for Maximum Independent Set of Hyperrectangles, where each update involves the insertion or deletion of a hyperrectangle. They presented a dynamic algorithm with polylogarithmic update time and a competitive ratio of $(1 + \varepsilon) \log^{d-1} K$, assuming that the coordinates of all input hyperrectangles are in $[0, K]^d$, and each of their edges has length at least 1. It is worth noting that, unlike dynamic algorithms, in online algorithms, decisions are irrevocable, and once hyperrrectangles are selected, they cannot be removed.

1.1 Our Contributions

In this work, we present a unified approach to design and analyze online algorithms that achieve *strong competitiveness* (high probability performance guarantees in the random-order arrival model). We elucidate our approach by studying the maximum independent set

[1] We show in the full version [37] that the requirement that the online algorithm knows the size of the instance n in the beginning, is important to obtain non-trivial results for our model.

[2] The *inductive independence number* ρ of a graph is the smallest number for which there exists an order $\prec$ such that for any independent set $U \subseteq V$ and any $v \in V$, we have $|\{u \in U | u \succ v \text{ and } \{u, v\} \in E\}| \leq \rho$.

problem across a variety of geometric objects, while the techniques introduced are possibly applicable to a variety of combinatorial problems. These algorithms are remarkably simple, and we demonstrate that, with the use of appropriate data structures, they operate in near-linear time. We obtain strongly $(\log n)^{O(1)}$-competitive algorithms for intervals, rectangles, d-dimensional hyperrectangles, fat objects, ellipsoids, and more.

First, we consider the case of intervals with integral endpoints lying in $[0, K]$ for some positive integer $K = n^{O(1)}$. Note that K, in general, can be quite large, e.g., $2^{\Theta(n)}$ as even then the input representation requires only bit size that is polynomial in n. Later, we will see how to get rid of this assumption on the value of K. We observe that, for intervals, if the intervals are of similar lengths, or if the instance is sparse (not too many intervals lie within any unit range, i.e., the underlying graph has a small maximum degree), then the greedy algorithm is $O(1)$-competitive. To leverage this observation, we categorize the intervals into $O(\log n)$ classes, ensuring that intervals in the same class have roughly similar lengths. The aim is to run the greedy algorithm on one of these classes, which has an independent set of size at least $\mathsf{opt}/O(\log n)$. Our algorithm begins with an initial *observation phase*, lasting for $n/2$ steps, at the end of which the algorithm picks a class that had the largest number of disjoint intervals. Subsequently, in the *action phase*, our algorithm greedily selects intervals arriving only from the identified class. With a loss of an additional $O(\log \log n)$-factor, we ensure that the optimum value in classes with large independent sets is evenly concentrated in both phases, with high probability, thus fulfilling our aim. The algorithm can be easily implemented in $O(n \log n)$ time using binary search trees. This leads to the following theorem.

▶ **Theorem 1.** *There exists a strongly $O(\log n \cdot \log \log n)$-competitive $O(n \log n)$-time algorithm for interval scheduling in ROM, where the intervals have integral endpoints in $[0, K]$ and $K = n^{O(1)}$.*

In fact, we demonstrate that our results hold for the general case of (K, D)-bounded instances (see Section 3 for the definition). Intuitively, this implies that the instance lies in $[0, K]$, and each unit range contains at most D intervals. We observe that for intervals with lengths at most 1, the greedy algorithm is $O(D)$-competitive. The observation becomes crucial for deciding which class to choose at the end of the observation phase; there is a trade-off between the optimal value available in a class and the loss in the competitive ratio that the greedy algorithm will incur on that class. Such instances later play an important role in removing the assumption on K.

Next, we show that our techniques can be generalized to hyperrectangles. One crucial difference, however, is that we cannot solve the independent set problem exactly for the various classes in the observation phase. Nevertheless, we show that we can still employ the greedy algorithm for estimating the optimal values for these classes. While a trivial implementation of the algorithm would take $\tilde{O}(n^2)$ time, the use of novel data structures allows us to improve the runtime to $O_d(n \log n)$ (see the full version [37] for details).[3] Intuitively, for each class, we maintain a uniform d-dimensional grid, where each grid cell stores all the hyperrectangles intersecting that cell. The grid cells are spaced in such a way that each cell is guaranteed to intersect only a constant number of hyperrectangles, and further, each hyperrectangle intersects only a constant number of grid cells. This ensures that the greedy algorithm only needs to examine $O(1)$ hyperrectangles (instead of $O(n)$) in order to decide whether or not to select an incoming hyperrectangle. We show that d binary search operations (one for each dimension) suffice to enumerate the aforesaid hyperrectangles, implying a total execution time of $O_d(n \log n)$ for our algorithm.

[3] $O_d(f(n))$ refers to a function in $O(f(n))$ when d is a constant.

▶ **Theorem 2.** *For all constants $d \geq 2$, there exists a strongly $(\log n)^{O(d)}$-competitive $O_d(n \log n)$-time algorithm for* Maximum Independent Set of d-dimensional Hyperrectangles *in ROM when the hyperrectangles have integral endpoints in $[0, K]^d$ and $K = n^{(\log n)^{O(1)}}$.*

Furthermore, we extend the results to σ-rectangular objects (see Definition 14). Intuitively, these objects can be made fat after some uniform scaling along the axes, and they encompass many important objects such as axis-aligned hyperrectangles, hyperspheres, fat objects, ellipsoids, etc.

▶ **Theorem 3.** *For all constants $d \geq 2$, there exists a strongly $(\sigma \log K)^{O(d)}$-competitive $O_d(n \log n)$-time algorithm for the maximum independent set problem in ROM for σ-rectangular objects in $[0, K]^d$ and $K = n^{(\log n)^{O(1)}}$.*

This provides an intriguing characterization of the problem; for thin objects, including arbitrary line segments, there is no known polynomial-time algorithm with a polylogarithmic approximation ratio, even in the offline setting.

Our bounds on competitive ratios above are indeed $(\log K)^{O(d)}$. Therefore, we assume K to be quasi-polynomially bounded to obtain a $(\log n)^{O(d)}$ guarantee. This assumption on the bounding box is common in many geometric problems, such as geometric knapsack [1], storage allocation problem [59], etc. For instance, the result of Henzinger et al. [45] for hyperrectangles in the dynamic setting holds under this assumption. Surprisingly, we are able to eliminate the assumption on K. For this, our algorithm has an initial *scale preparation* phase lasting for about $n/2$ steps to learn about the instances and project $[0, K]$ to $[0, n]$. Using hypergeometric concentration inequalities, we show that our projection creates a (K, D)-bounded instance with $K = O(n)$ and $D = O(\log n)$.

Roughly speaking, the key observation that facilitates such a projection is as follows. Let L denote the set of all left endpoints of the intervals in the input instance and $\hat{L} \subseteq L$ denote the set of those left endpoints observed during the scale preparation phase. Then, $\hat{L}$ forms a random subset of L of size $n/2$ selected uniformly at random. Consequently, the left endpoints in $\hat{L}$ are evenly distributed among the left endpoints in L. In particular, between any two consecutive elements of $\hat{L}$ (when it is sorted), at most $O(\log n)$ elements of $L \setminus \hat{L}$ appear with high probability. In the subsequent non-uniform scale that we prepare, the left endpoints of $\hat{L}$ are mapped to $\{1, 2, \ldots, n/2\}$, while the remaining points on the real line are projected appropriately, maintaining their original ordering. When the intervals are read in this non-uniform scale, at most $O(\log n)$ intervals are contained within $(i, i+1)$ for all $i < n/2$ with high probability. Thus, the original input instance when viewed under such a non-uniform scale appears to be a (K, D)-bounded instance for $K = O(n)$ and $D = O(\log n)$.

Our technique of mapping $[0, K]$ to $[0, n]$ might prove valuable in other related problems (e.g., problems related to geometric intersection graphs of objects confined within a bounding box of arbitrary size). Additionally, we crucially leverage the fact that axis-aligned rectangles remain axis-aligned rectangles under non-uniform scaling along the axes. We then apply our previous algorithms as subroutines and obtain the following theorems.

▶ **Theorem 4.** *There exists a strongly $O(\log n \cdot \log \log n)$-competitive $O(n \log n)$-time algorithm for interval scheduling in ROM.*

▶ **Theorem 5.** *For all constants $d \geq 2$, there exists a strongly $O((\log n)^d \cdot \log \log n)$-competitive $O_d(n \log n)$-time algorithm for* Maximum Independent Set of d-dimensional Hyperrectangles *in ROM.*

Note that we only consider the unweighted setting. High probability guarantees are not possible in the weighted setting as even in the special case of the *Secretary problem* (when all intervals have the same start and endpoints, but may have different weights), the probability of success is at most $1/e$ [24].

In summary, we introduce a unified framework designed to achieve strongly competitive online algorithms in the random-order model when dealing with geometric objects confined within a bounded box (which may be of arbitrary size). As any permutation of the objects is equally likely, the underlying structure and arrangement of the objects are effectively captured within a constant fraction of the input. Consequently, with a negligible $O(1)$ loss in the competitive ratio, the size of the bounding box can be reduced to $K = n^{O(1)}$. Following this reduction, the input is carefully decomposed into $k = (\log n)^{O(1)}$ classes, each class i admitting a nice α_i-competitive algorithm. The effective balance of these α_i's results in an overall competitive ratio smaller than $\tilde{O}(k \cdot \max \alpha_i)$ for the original problem. For different problems, the ingenuity lies in devising a clever decomposition into classes, ensuring effective approximation of the classes and a balanced distribution of the approximation factors. Our algorithms for Maximum Independent Set of Hyperrectangles are not only simple and fast but also nearly match the best offline approximation guarantees, falling short by only a $\log^2 n$ factor.

1.2 Further Related Work

The study of online interval scheduling was initiated by Lipton and Tomkins [55], who presented an $O((\log \Delta)^{1+\varepsilon})$-competitive algorithm when intervals arrive in the order of their start times, where Δ is the ratio of the lengths of the longest and shortest intervals in the input. However, due to the $\Omega(n)$ lower bound in the adversarial-order arrival model, online interval scheduling has only been explored in restricted settings [61, 31, 60, 63, 13, 14, 8, 62].

Recently, De, Khurana, and Singh [22] studied the online independent set and online dominating set problems, presenting an algorithm with a competitive ratio equal to the independent kissing number of the corresponding geometric intersection graph.[4] However, these results do not provide meaningful bounds for intervals or hyperrectangles, as considered in our work. Building on the work of Henzinger et al. [45], Bhore et al. [11] investigated the maximum independent set problem in the fully dynamic setting for various classes of geometric objects, including intervals and d-dimensional hypercubes. They showed that a $(1+\varepsilon)$-approximation can be maintained in logarithmic update time for intervals, which generalizes to an $O(4^d)$-competitive algorithm with polylogarithmic update time for hypercubes.

The Maximum Independent Set of Hyperrectangles is closely related to many fundamental problems in geometric approximation, such as geometric knapsack [33, 46, 34], geometric bin packing [9, 51], strip packing [32, 49], etc. For more results on multidimensional packing and covering problems, see [18]. Various geometric problems have been extensively studied in the online setting, including geometric set cover [50], hitting set [26], piercing set [21] etc. The streaming model has also received some attention [15, 12, 20]. Additionally, the random-order arrival model has been explored for classical problems such as knapsack [6, 48], bin packing [47, 5, 7, 44], set cover [39], matroid intersection [41], matching [29], and more.

[4] For a family of geometric objects $\mathcal{S}$, the independent kissing number is defined as the maximum number of pairwise non-touching objects in $\mathcal{S}$ that can be arranged in and around a particular object such that all of them are dominated/intersected by that object.

Organization of the paper. In Section 2, we discuss preliminaries. Section 3 deals with intervals contained within a bounding box of size polynomial in n and we prove Theorem 1. We then generalize this result to hyperrectangles and other objects in Section 4, proving Theorems 2 and 3. Section 5 contains the proofs for intervals and hyperrectangles confined within arbitrary sized bounding box: Theorems 4 and 5. Finally, Section 6 contains some concluding remarks. Due to space limitations, some parts have been moved to the full version of the paper [37].

2 Preliminaries

For all positive integers n, let $[n] := \{1, 2, \dots, n\}$. We will assume the base of log is e, unless otherwise specified. For sets A and B, $A \uplus B$ denotes their disjoint union. For an interval I, let $||I||$ denote its length. For the case of hyperrectangles, we will assume $\mathcal{H} := \{H_1, H_2, \dots, H_n\}$ to be the input set of n d-dimensional hyperectangles, where $H_i := [x^1_{i,1}, x^2_{i,1}] \times [x^1_{i,2}, x^2_{i,2}] \times \cdots \times [x^1_{i,d}, x^2_{i,d}]$. Here, $0 \le x^1_{i,j} \le x^2_{i,j} \le K$ for all $i \in [n], j \in [d]$, and the length of H_i along dimension j is $l_j(H_i) := x^2_{i,j} - x^1_{i,j}$. Two objects H_i and H_j *intersect* if $H_i \cap H_j \neq \emptyset$. Let $\mathsf{OPT}(\mathcal{H})$ denote a maximum independent set for the intersection graph corresponding to $\mathcal{H}$. The Maximum Independent Set of Hyperrectangles problem aims to determine $\mathsf{OPT}(\mathcal{H})$. Let $\mathsf{opt}(\mathcal{H}) := |\mathsf{OPT}(\mathcal{H})|$ be the size of an optimal solution.

The following lemma will be crucial in our analysis (see the full version [37] for a proof).

▶ **Lemma 6** (Hypergeometric concentration). *There are N balls of which M are red. Some n balls are sampled from the N balls uniformly at random without replacement. Let X be the number of red balls that appear in the sample. Let $p = \frac{M}{N}$ and $0 \le \delta \le 1$. Then,*

$$\Pr[X \ge (1+\delta)pn] \le \exp(-\delta^2 pn/3); \ \Pr[X \le (1-\delta)pn] \le \exp(-\delta^2 pn/2).$$

2.1 Greedy Algorithm

Throughout the paper, we let Greedy represent the greedy algorithm, which selects an object on arrival if it does not intersect previously selected objects. Clearly, Greedy returns a maximal independent set of the geometric intersection graph. We show in the full version [37] that the competitive ratio of Greedy is at least $\Omega(\sqrt{n})$ in the random-order model.

▶ **Theorem 7.** Greedy *has a competitive ratio of at least $\Omega(\sqrt{n})$ in the random-order model.*

Next, we state two properties of Greedy that we shall frequently use.

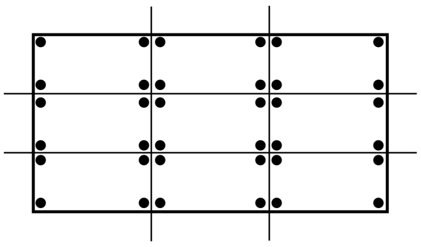


Figure 1 For $d = 2$, $\Delta = 3$, a piercing set of $\mathcal{H}$ can be formed by placing 36 points inside each $\mathcal{H}'$.

The first property states that Greedy performs well when the input instance consists of hyperrectangles of similar size.

▶ **Lemma 8.** *Let $\mathcal{H}$ be a set of hyperrectangles such that for any $H_1, H_2 \in \mathcal{H}$ and for any dimension $j \in [d]$, we have $l_j(H_1)/l_j(H_2) \in [1/\Delta, \Delta]$ for some integer Δ. Let $\mathcal{H}'$ be any maximal independent subset of hyperrectangles from $\mathcal{H}$. then* $\mathsf{opt}(\mathcal{H}) \leq (2\Delta)^d \cdot |\mathcal{H}'|$.

Proof. Let P^* be the minimum piercing set of $\mathcal{H}$, i.e., a set of points of minimum size such that each hyperrectangle in $\mathcal{H}$ is stabbed by at least one point in P^*. Then clearly $\mathsf{opt}(\mathcal{H}) \leq |P^*|$, since no two hyperrectangles in $\mathsf{opt}(\mathcal{H})$ can be stabbed by the same point in P^*. We next show that $|P^*| \leq (2\Delta)^d \cdot |\mathcal{H}'|$, thus completing the proof. Consider any $H \in \mathcal{H}'$. We divide H into Δ^d identical homothetic copies using $\Delta - 1$ equally-spaced axis-aligned hyperplanes in each dimension (see Figure 1). Then observe that any hyperrectangle that intersects H must contain a corner of at least one of these smaller hyperrectangles because of the assumption on the ratio of the side lengths. Since $\mathcal{H}'$ is maximal, it follows that the set of $(2\Delta)^d$ corners of these Δ^d hyperrectangles form a piercing set of the hyperrectangles intersecting H. Hence $|P^*| \leq (2\Delta)^d \cdot |\mathcal{H}'|$. ◀

The next lemma is based on the fact that Greedy performs satisfactorily when the maximum degree of the underlying intersection graph is small.

▶ **Lemma 9.** *Let $\mathcal{H}$ be a set of hyperrectangles such that $l_j(H_i) \leq 1$ for all $H_i \in \mathcal{H}$ in some dimension $j \in [d]$. Also suppose for each integer $m \in [K]$, the number of hyperrectangles H_i with $x^1_{i,j} \in (m-1, m)$ is at most D, where $D \in \mathbb{N}$. Then* $\mathsf{opt}(\mathcal{H}) \leq 3D \cdot |\mathsf{Greedy}(\mathcal{H})|$.

Proof. The conditions of the lemma imply that each hyperrectangle in $\mathcal{H}$ can overlap with at most $3D - 1$ other hyperrectangles. Therefore, the maximum degree of a node in the intersection graph of $\mathcal{H}$ is at most $3D - 1$. It is a folklore result that the greedy algorithm selects an independent set of size at least $n/(\Delta + 1)$ in any n-vertex graph with maximum degree Δ. Therefore $\mathsf{opt}(\mathcal{H}) \leq 3D \cdot |\mathsf{Greedy}(\mathcal{H})|$. ◀

3 Online Algorithms For Intervals

In this section, we prove Theorem 1. We first prove the following general lemma for *bounded* instances, which are defined as follows.

▶ **Definition 10.** *A set of intervals $\mathcal{I}$ is* (K, D)-bounded *if each $I \in \mathcal{I}$ is a subset of $[0, K]$ and for all $i \in \mathbb{Z}$, the number of intervals from $\mathcal{I}$ that are a subset of $(i, i+1)$ is at most D.*

We show that there exists an online algorithm which on (K, D)-bounded instances is $O(D + \log K \cdot \log\log K)$-competitive with probability $1 - o_K(1)$.

▶ **Lemma 11.** *There exists absolute constants $\lambda \in (0, 1)$, $\mu \geq 1$, and an online algorithm such that for each (K, D)-bounded instance $\mathcal{I}$, where K and $n := |\mathcal{I}|$ are large enough, the algorithm outputs an independent set from $\mathcal{I}$ of size at least $\frac{\lambda}{D + \log K \cdot \log\log K} \cdot \mathsf{opt}(\mathcal{I})$ with probability at least $1 - \frac{\mu}{\log K}$.*

Proof. Let $\varepsilon := 1$ and $k := \lceil \log_{1+\varepsilon} K \rceil$. We partition $\mathcal{I}$ into $k+1$ sets: $S_0, \ldots, S_k$, depending on the lengths of the intervals in $\mathcal{I}$, such that $S_0 := \{I \in \mathcal{I} \mid ||I|| \in [0, 1]\}$ and $S_i := \{I \in \mathcal{I} \mid ||I|| \in ((1+\varepsilon)^{i-1}, (1+\varepsilon)^i]\}$ for all $i \in [k]$. We now describe our algorithm before moving to its analysis.

Algorithm. The algorithm has two phases: the *observation* phase, when it receives the first $\lceil \frac{n}{2} \rceil$ intervals (we call them $\mathcal{I}_0$) and the *action* phase, when it receives the remaining $n - \lceil \frac{n}{2} \rceil$ intervals (we call them $\mathcal{I}_1$). Note that $\mathcal{I} = \mathcal{I}_0 \uplus \mathcal{I}_1$, where $\mathcal{I}_0$ is a uniformly random subset of $\mathcal{I}$ of size $\lceil \frac{n}{2} \rceil$, and $\mathcal{I}_1$ is a uniformly random subset of $\mathcal{I}$ of size $n - \lceil \frac{n}{2} \rceil$.

- **Observation phase** In this phase intervals in $\mathcal{I}_0$ arrive.
  - Initialization: Initialize $L_i = \emptyset$ for $i \in \{0\} \cup [k]$.
  - When a new interval $I \in \mathcal{I}_0$ arrives, add I to L_i iff $I \in S_i$.
  - At the end of the observation phase, all L_i's are such that $L_i = \mathcal{I}_0 \cap S_i$. Compute $\mathsf{opt}(L_i)$ for each $i \in \{0\} \cup [k]$.
  - Compute the index $m \in \{0\} \cup [k]$ as follows. Let $m \in [k]$ be an index such that $\mathsf{opt}(L_m) \geq \mathsf{opt}(L_i)$ for all $i \in [k]$. If $\mathsf{opt}(L_0) > k \cdot \mathsf{opt}(L_m)$, set $m = 0$.
- **Action phase** In this phase, the intervals in $\mathcal{I}_1$ arrive.
  - Initialization: $R_m = \emptyset$. On arrival of an interval $I \in \mathcal{I}_1$, do the following.
  - If $I \in S_m$ and I does not intersect any intervals in R_m select I and add I to R_m.
  - Else, if I is the last interval to arrive and $R_m = \emptyset$, then select I.
  - Else, if $I \notin S_m$, then do not select I.

Thus, in the observation phase, the algorithm computes $\mathsf{opt}(L_i)$, where $L_i = S_i \cap \mathcal{I}_0$, for all $i \in 0 \cup [k]$. Then, it sets $m = 0$ when $\mathsf{opt}(L_0) > k \cdot \mathsf{opt}(L_i)$ for all $i \in [k]$, else it sets $m \in [k]$ such that $\mathsf{opt}(L_m) \geq \mathsf{opt}(L_i)$ for all $i \in [k]$. In the action phase, the algorithm simply runs the greedy algorithm Greedy on $R_m := \mathcal{I}_1 \cap S_m$, ignoring all intervals not in S_m. When the algorithm receives the very last interval and $R_m = \emptyset$, it picks the last interval. This ensures that the algorithm outputs an independent set of size at least one whenever $n \geq 2$.

Analysis. We now analyze the competitive ratio of the above algorithm.

$$\text{We may assume that } \mathsf{opt}(\mathcal{I}) \geq 10^4 k \log k. \tag{1}$$

Otherwise, since the algorithm is guaranteed to pick at least one interval (for $n \geq 2$), we get a competitive ratio of $10^4 k \log k$.

We now define an event E and show that (I) if E holds, then the algorithm achieves a competitive ratio of $O(D + K)$, and (II) E holds with probability $1 - o_k(1)$. In order to define the event E, we first define $L_i = \mathcal{I}_0 \cap S_i$ and $R_i = \mathcal{I}_1 \cap S_i$ for all $i \in \{0\} \cup [k]$, as in the algorithm above. Furthermore, let us define $B = \{i \in \{0\} \cup [k] \mid \mathsf{opt}(S_i) \geq 10^3 \log k\}$.

We say that the event E holds iff for all $i \in B$,

$$\min(\mathsf{opt}(L_i), \mathsf{opt}(R_i)) \geq (1 - \delta)\frac{\mathsf{opt}(S_i)}{2}, \quad \text{where } \delta = \frac{1}{10}. \tag{2}$$

(I) Assuming E holds, the algorithm is $O(D + k)$-competitive. We prove two basic bounds that will help us in the proof. First, we want to show that $\sum_{i \in B} \mathsf{opt}(S_i)$ is large, i.e., at least a constant fraction of $\mathsf{opt}(\mathcal{I})$. Intuitively, this would imply that even if an algorithm ignored intervals in $\cup_{i \notin B} S_i$, it would have substantial value available.

$$\mathsf{opt}(\mathcal{I}) \leq \sum_{i \in B} \mathsf{opt}(S_i) + \sum_{i \notin B} \mathsf{opt}(S_i) \implies \frac{\mathsf{opt}(\mathcal{I})}{2} \leq \sum_{i \in B} \mathsf{opt}(S_i), \tag{3}$$

where the last inequality holds as $\sum_{i \notin B} \mathsf{opt}(S_i) \leq (k+1) \cdot 10^3 \log k \leq 5 \cdot 10^3 k \log k \leq \frac{\mathsf{opt}(\mathcal{I})}{2}$. Next, we want to show for $i \in B$, $\mathsf{opt}(R_i)$ is at least a constant fraction of $\mathsf{opt}(L_i)$.

$$\text{For each } i \in B,\ \mathsf{opt}(R_i) \geq \frac{1 - \delta}{2}\mathsf{opt}(S_i) \geq \frac{1 - \delta}{2}\mathsf{opt}(L_i), \tag{4}$$

where the last inequality holds as $L_i \subseteq S_i$. We now prove the following claim.

▷ Claim 12. Let m be the index chosen at the end of the observation phase, then $m \in B$, Proof of claim. We assume $m \notin B$ and derive a contradiction. From Bounds (2) and (3):

$$\sum_{i\in B} \mathsf{opt}(L_i) \geq \frac{1-\delta}{4}\mathsf{opt}(\mathcal{I}) \geq \frac{(1-\delta)\cdot 10^4}{4} k\log k = \frac{9}{4}\cdot 10^3 \cdot k\log k. \tag{5}$$

Whereas, if $m \notin B$, then

$$\sum_{i\in B} \mathsf{opt}(L_i) \leq \sum_{i=0}^{k} \mathsf{opt}(L_i) \leq k\cdot 10^3 \log k + k\cdot 10^3\log k = 2\cdot 10^3\cdot k\log k \tag{6}$$

The last inequality follows from the facts that for all $i \in [k]$, $\mathsf{opt}(L_i) \leq \mathsf{opt}(L_m)$, $\mathsf{opt}(L_0) \leq k\cdot\mathsf{opt}(L_m)$, and as $m \notin B$ we have $\mathsf{opt}(L_m) \leq \mathsf{opt}(S_m) \leq 10^3\log k$. Thus, from inequalities (5) and (6), we get a contradiction. Thus, $m \in B$. ◁

We now have two cases depending on whether the algorithm picks $m = 0$ or $m \neq 0$ at the end of the observation phase. In either case, we show $|\mathsf{Greedy}(R_m)| \geq \frac{1}{O(D+k)}\mathsf{opt}(\mathcal{I})$.

Case: $m = 0$.

$$\begin{aligned}\frac{\mathsf{opt}(\mathcal{I})}{2} &\leq \sum_{i\in B}\mathsf{opt}(S_i) \leq \frac{2}{1-\delta}\sum_{i\in B}\mathsf{opt}(L_i)\\ &\leq \frac{2}{1-\delta}\left(\mathsf{opt}(L_0) + \sum_{i\in B\setminus\{0\}}\mathsf{opt}(L_i)\right)\\ &\leq \frac{2}{1-\delta}\left(\mathsf{opt}(L_0) + k\cdot\frac{1}{k}\cdot\mathsf{opt}(L_0)\right),\end{aligned}$$

where the first and second inequalities follow from Bounds (3), (2), the last inequality follows from the fact that the algorithm picks $m = 0$ when $\mathsf{opt}(L_0) \geq k\cdot\mathsf{opt}(L_i)$ for all $i \in [k]$. Thus,

$$\mathsf{opt}(L_0) \geq \frac{1-\delta}{8}\mathsf{opt}(\mathcal{I}) \implies \mathsf{opt}(R_0) \geq \frac{(1-\delta)^2}{16}\mathsf{opt}(\mathcal{I}) \quad \text{(from Bound (4)).}$$

Now from Lemma 9, the output size $|\mathsf{Greedy}(R_0)| \geq \frac{1}{3D}\mathsf{opt}(R_0) \geq \frac{(1-\delta)^2}{48D}\mathsf{opt}(\mathcal{I})$. (7)

Case: $m \neq 0$.

$$\frac{\mathsf{opt}(\mathcal{I})}{2} \leq \frac{2}{1-\delta}\left(\mathsf{opt}(L_0) + \sum_{i\in B\setminus\{0\}}\mathsf{opt}(L_i)\right) \leq \frac{2}{1-\delta}\left(k\cdot\mathsf{opt}(L_m) + k\cdot\mathsf{opt}(L_m)\right),$$

where the first inequality follows the same way as in case $m = 0$ above; the last follows from the fact that $\mathsf{opt}(L_0) \leq k\cdot\mathsf{opt}(L_m)$ as $m \neq 0$ and $\mathsf{opt}(L_i) \leq \mathsf{opt}(L_m)$ for all $i \in [k]$. Thus,

$$\mathsf{opt}(L_m) \geq \frac{1-\delta}{8k}\mathsf{opt}(\mathcal{I}) \implies \mathsf{opt}(R_m) \geq \frac{(1-\delta)^2}{16k}\mathsf{opt}(\mathcal{I}) \quad \text{(from Bound (4)).}$$

Now from Lemma 8, the output size $|\mathsf{Greedy}(R_m)| \geq \frac{1}{4}\mathsf{opt}(R_m) \geq \frac{(1-\delta)^2}{64k}\mathsf{opt}(\mathcal{I})$. (8)

Thus, in either case, from Bounds (7) and (8), $|\mathsf{Greedy}(R_m)| \geq \frac{(1-\delta)^2}{64(D+k)}\mathsf{opt}(\mathcal{I})$. (9)

Thus, from Bounds (1) and (9), our algorithm is $O(\max(k\log k, D+k))$-competitive, which implies that it is $O(D + k\log k)$-competitive. We now show that (II) is true.

(II) E holds with probability at least $1 - 3/k$. Fix an $i \in B$. In the following, we use the fact that $\mathcal{I}_0$ is a random subset of $\mathcal{I}$ of size $\lceil \frac{n}{2} \rceil$, and upper bound the probability that only a small fraction of $\mathsf{OPT}(S_i)$ is picked in $\mathcal{I}_0$.

$$\begin{aligned}
&\Pr\left[\mathsf{opt}(L_i) < (1-\delta)\frac{\mathsf{opt}(S_i)}{2}\right] \leq \Pr\left[\mathsf{opt}(L_i) \leq (1-\delta)\frac{\mathsf{opt}(S_i)}{n}\left\lceil\frac{n}{2}\right\rceil\right] \\
&\leq \exp\left(-\frac{\delta^2}{2}\frac{\mathsf{opt}(S_i)}{n}\left\lceil\frac{n}{2}\right\rceil\right) \leq \exp\left(-\frac{\delta^2}{4}\mathsf{opt}(S_i)\right) \leq \exp\left(-\frac{10^3\delta^2}{4}\log k\right) \leq \frac{1}{k^2},
\end{aligned}$$

where the second inequality follows from Lemma 6 where N, M, P, and n in the Lemma statement are set to $n, \mathsf{opt}(S_i), \frac{\mathsf{opt}(S_i)}{n}$, and $\lceil \frac{n}{2} \rceil$, respectively, and the fourth inequality follows since $\mathsf{opt}(S_i) \geq 10^3 \log k$, as $i \in B$.

A similar calculation, where we use the fact that $\mathcal{I}_1$ is a uniformly random subset of $\mathcal{I}$ of size $n - \lceil \frac{n}{2} \rceil$, yields for all $n \geq 100$, $\Pr\left[\mathsf{opt}(R_i) < (1-\delta)\frac{\mathsf{opt}(S_i)}{2}\right] \leq \frac{1}{k^2}$.

Now, from the union bound, $\Pr[\exists i \in B \text{ such that } \min(\mathsf{opt}(L_i), \mathsf{opt}(R_i)) < (1-\delta)\frac{\mathsf{opt}(S_i)}{2}] \leq \frac{2|B|}{k^2} \leq \frac{2k+2}{k^2} \leq \frac{3}{k}$, for all $k \geq 2$. Thus, $\Pr[E] \geq 1 - \frac{3}{k}$. This completes the proof. ◀

Now, when the intervals have integral endpoints in $[0, K]$, where $K = n^{O(1)}$, then $D = 0$. Plugging these values in the statement of Lemma 11, we obtain Theorem 1.

4 Generalization to Hyperrectangles and Other Objects

In this section, we first prove Theorem 2. We extend our result obtained for intervals to the maximum independent set of (K, D)-bounded set of hyperrectangles (rigorously defined in the full version [37]). The input instance $\mathcal{H}$ consists of hyperrectangles that are in the bounding box $[0, K]^d$, and for each $i \in [K]$ and $j \in [d]$ the number of hyperrectangles in $\mathcal{H}$ whose projections along the jth axis falls in the interval $(j, j+1)$ is at most D. We prove the following general lemma.

▶ **Lemma 13.** *For each $d \in \mathbb{N}$, there exist constants $\lambda \in (0, 1)$, $\mu \geq 1$, and an online algorithm such that for each (K, D)-bounded set of d-dimensional hyperrectangles $\mathcal{H}$ where K and $|\mathcal{H}|$ are large enough, the algorithm outputs an independent set from $\mathcal{H}$ of size at least $\frac{\lambda}{D^2 + (\log K)^d \cdot \log\log K} \cdot \mathsf{opt}(\mathcal{H})$ with probability at least $1 - \frac{\mu}{\log K}$.*

We refer the reader to the full version [37] for a complete proof of Lemma 13. The proof is similar to that of Lemma 11, with the algorithm having observation/action phases, with three notable differences:

- **Partitioning:** We partition the input instance $\mathcal{H}$ into $d + (\log K)^d$ classes: S_x's for $x \in [d]$ and S_y's for $y \in [\log K]^d$, based on the side lengths of the hyperrectangles in $\mathcal{H}$. Each S_x has hyperrectangles H such that $l_x(H) \leq 1$ (Greedy is $O(D)$-competitive for such instances). Each S_y has hyperrectangles that have similar lengths in each dimension (Greedy is 4^d-competitive for such instances). Like in the case of intervals, $S_i = L_i \uplus R_i$, where L_i are hyperrectangles in S_i that arrive during the observation phase.
- **Estimating:** Calculating $\mathsf{opt}(L_i)$ is hard; computing a maximum independent set is NP-complete even for unit squares [27]. Thus, we estimate their sizes with Greedy: $\hat{L}_i = \mathsf{Greedy}(L_i)$.

- **Balancing:** Similar to the interval case, we shall select only a particular size class m based on the estimates of opt values seen during the observation phase. Due to the difference in the competitiveness achieved by Greedy on S_x's and S_y's, we need to balance the estimates while choosing m. Let $m_1 = \arg\max_x \hat{L}_x$ and $m_2 = \arg\max_y \hat{L}_y$. If $\hat{L}_{m_1} \geq \frac{(k+1)^d}{D} \hat{L}_{m_2}$ set $m = m_1$, else set $m = m_2$. As before the algorithm outputs $\mathsf{Greedy}(R_m)$.

For the special case of $K = n^{(\log n)^{O(1)}}$ and the input hyperrectangles having integer coordinates, i.e., $D = 0$, Lemma 13 implies Theorem 2.

In the full version [37], we show how to extend our results to Ellipses and Fat objects, proving Theorem 3. In fact our results hold for σ-rectangular objects which generalize ellipses, equilateral triangles, fat objects, etc.

▶ **Definition 14.** *An object F is said to be σ-rectangular ($\sigma > 1$) if there exist axis-aligned inscribing and circumscribing hyperrectangles of F: $\mathsf{In}(F)$ and $\mathsf{Out}(F)$, respectively, such that for all $j \in [d]$, $l_j(\mathsf{In}(F)) \geq l_j(\mathsf{Out}(F))/\sigma$.*

Note that many common geometric objects have small values of σ (see the full version [37]). Unlike rectangles, these objects may not preserve their structural properties (fatness, spherical shape, or even convex shapes) under nonuniform scaling. Thus we cannot use our scale preparation steps directly. However, in the full version [37] we show that we can still handle them and prove Theorem 3.

5 Removal of the Assumption on K

In this section, we show how to get rid of the dependence on K in the competitive ratio. We shall first consider the case of intervals and prove Theorem 4 using Lemma 11. A key ingredient in our proof will be the following lemma which, roughly speaking, states that if we pick a random set $T \subseteq [n]$ of size $n/2$ uniformly at random, the $n/2+1$ *gaps* induced by T on $[n]$ are each at most $4\log n$ with high probability.

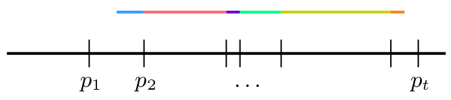
▶ **Lemma 15.** *Let T be a subset of $[n]$ of cardinality $\lceil \frac{n}{2} \rceil$ chosen uniformly at random. Let $T = \{X_i \mid X_i \in [n], i \in [\lceil n/2 \rceil]\}$, where $1 = X_0 \leq X_1 < X_2 < \cdots < X_{\lceil \frac{n}{2} \rceil} \leq X_{\lceil \frac{n}{2} \rceil + 1} = n$.*

$$\text{Then,} \quad \mathsf{Pr}\left[\max_{i \in [\lceil \frac{n}{2} \rceil + 1]} (X_i - X_{i-1}) \leq 4\lceil \log_2 n \rceil \right] \geq 1 - \frac{1}{n}.$$

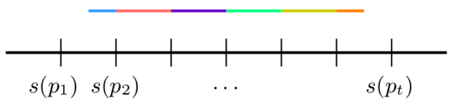
See the full version [37] for the formal proof of this lemma. We briefly explain the intuition behind the proof. Roughly speaking, we divide $[n]$ into contiguous blocks of length $2\log_2 n$ each. Observe that if T hits all such blocks, the gaps induced by T on $[n]$ are each of length at most $4\log_2 n$. Now, since each element of $[n]$ is being included in T with probability $1/2$, we can argue that the probability that T does not intersect a fixed block is at most $1/(2^{2\log_2 n}) = \frac{1}{n^2}$. Applying a union bound over all such blocks, which are at most n in number, we have that T must hit all blocks with probability at least $1 - \frac{1}{n}$.

We now prove Theorem 4.

Proof of Theorem 4. Our idea is the following. We will divide the algorithm into two phases, each consisting of about $n/2$ intervals. In the first phase, we will not pick any of the intervals. Based on the intervals that arrived in the first phase, we will do a *non-uniform scaling* of the real line, so that the intervals arriving in the second phase when read in this scale will form a (K, D)-bounded instance for $K = O(n)$ and $D = O(\log n)$. In the second phase, we will run the algorithm in Lemma 11 to get the desired competitive ratio.



(a) Original interval



(b) After scaling

Figure 2 Non-uniform scaling. The vertical bars on the axis represent the t starting points of the intervals in $\mathcal{I}_0$. After scaling, they become evenly spaced.

Let $\mathcal{I}$ be the input instance with $|\mathcal{I}| = n$. Let $\mathcal{I} = \mathcal{I}_0 \uplus \mathcal{I}_1$, where I_0 consists of the intervals received in the first phase, where $|I_0| = \lceil \frac{n}{2} \rceil$, and I_1 consists of the intervals received in the second phase.

Non-uniform Scaling. We first see how to obtain the desired non-uniform scaling. For an interval I, let $\ell(I)$ denote its left endpoint. Let $I_1, \cdots, I_n$ be all the intervals in $\mathcal{I}$ such that $\ell(I_1) \leq \cdots \leq \ell(I_n)$. Note that the $\ell(I_j)$'s need not be all distinct. Observe that the set of subscripts of the intervals in $\mathcal{I}_0$ is a random subset of $[n]$ of size $\lceil \frac{n}{2} \rceil$ chosen uniformly at random. Let $\mathcal{I}_0 = \{I_{X_1}, \cdots, I_{X_{\lceil n/2 \rceil}}\}$ where $1 = X_0 \leq X_1 < \cdots < X_{\lceil n/2 \rceil} \leq X_{\lceil n/2 \rceil + 1} = n$. Let ξ_1 be the event $\max_{i \in [\lceil n/2 \rceil + 1]}(X_i - X_{i-1}) \leq 4\lceil \log_2 n \rceil$. Lemma 15 yields that ξ_1 holds with probability $1 - 1/n$. Since I_j's are ordered by their left endpoints, if the event ξ_1 holds, then for each $i \in [\lceil n/2 \rceil + 1]$, there are at most $4\lceil \log_2 n \rceil$ intervals in $\mathcal{I}$ whose left endpoints are between $\ell(I_{X_{i-1}})$ and $\ell(I_{X_i})$. Let $\{p_1, \cdots, p_t\} = \{\ell(I_{X_1}), \cdots, \ell(I_{X_{\lceil n/2 \rceil}})\}$ be the set of t left endpoints of the intervals in $\mathcal{I}_0$, where $t \leq \lceil n/2 \rceil$ (we do not necessarily have an equality as some of the left endpoints might be the same) such that $p_1 < \cdots < p_t$. In the second phase, we will not select any interval I whose left endpoint is to the left of p_1 or right of p_t, i.e., $\ell(I) < p_1$ or $\ell(I) > p_t$; let $F \subseteq \mathcal{I}$ denote all such intervals that need to be ignored. From the above discussion, if ξ_1 holds, then $|F| \leq 8\lceil \log_2 n \rceil$.

We now define our new scale $s : [p_1, \infty) \to [1, t]$ as follows.

$$s(x) = \begin{cases} i & \text{if } x = p_i \text{ for some } i \in [t] \\ i + \frac{x - p_i}{p_{i+1} - p_i} & \text{if } p_i < x < p_{i+1} \text{ for some } i \in [t-1] \\ t & \text{if } x > p_t \end{cases}$$

In the second phase, each interval in $\mathcal{I}_1 \setminus F$, will be read in the scale s, i.e., the interval $[a, b]$ will be read as $[s(a), s(b)]$. See Figure 2 for an example. The function s essentially scales the set $[0, K]$ such that the t left endpoints of the intervals in $\mathcal{I}_0$ become equispaced. Figure 2(b) illustrates how an interval in $\mathcal{I}_1 \setminus F$ looks after the scaling operation. Additionally, intervals in F will be all set to $[0, 0]$. Observe that this scaling does not change the underlying intersection graph induced on $\mathcal{I}_1 \setminus F$. Also, notice that for all $[a, b] \in \mathcal{I}_1 \setminus F$, $[s(a), s(b)] \subseteq [1, t] \subseteq [0, \lceil n/2 \rceil]$. For each $i \in \{0\} \cup [t-1]$, let D_i denote the number of intervals (read in the scale s if they are not in F and read as $[0, 0]$ if they are in F) from $\mathcal{I}_1$ that are contained in $(i, i+1)$. Let $D = \max_{i \in \{0\} \cup [t-1]} D_i$. If ξ_1 holds, then $D \leq 4\lceil \log_2 n \rceil$ since an interval is contained in $(i, i+1)$ only if its left endpoint is contained in $(i, i+1)$. Thus, at the end of the first phase, if ξ_1 holds, we will end up with a (K, D)-bounded instance ($\mathcal{I}_1$ when scaled as explained), where $K = \lceil n/2 \rceil$ and $D \leq 4\lceil \log_2 n \rceil$, on which we shall then run the algorithm in Lemma 11.

Algorithm. We now describe our algorithm. Let Alg denote the algorithm in Lemma 11.

- **First phase: Scale preparation.** In this phase, the intervals in $\mathcal{I}_0$ arrive. Let $p_1 < \cdots < p_t$ be the distinct left endpoints of the intervals in $\mathcal{I}_1$. Define the scale s as described above.

- **Second phase: Outsourcing.** In this phase, the intervals from $\mathcal{I}_1$ arrive. We will feed these intervals to the algorithm in Lemma 11, namely Alg, after appropriately scaling them with s. First, we feed Alg with the size of the instance it will receive: $n - \lceil n/2 \rceil$. On receiving the interval $[a, b] \in \mathcal{I}_1$,
  - If $[a, b]$ is the last interval and no interval has been selected before, select $[a, b]$.
  - Else, if $[a, b] \in F$, i.e., $a < p_1$ or $a > p_t$, we feed Alg the interval $[0, 0]$, but do not select $[a, b]$ even if Alg selects $[0, 0]$.
  - Else, feed the interval $[s(a), s(b)]$ to Alg and select the interval $[a, b]$ iff $[s(a), s(b)]$ is selected by Alg.

Analysis. Note that the intervals selected by the algorithm form an independent set. Let h be the size of this independent set. Observe that, if we condition on $\mathcal{I}_1$ being a particular $(K = \lceil n/2 \rceil, D)$-bounded instance and F being a particular set, then from Lemma 11, we conclude that for all large K and n, the above algorithm will output an independent set of size $h \geq \lambda \cdot \mathsf{opt}(\mathcal{I}_1 \setminus F)/(D + \log K \log \log K) - 1$ with probability at least $1 - \mu/\log K$, where $\lambda \in (0, 1)$ and $\mu \geq 1$ are absolute constants. The -1 term appears in the above guarantee because Alg might select the interval $[0, 0]$ fed to it corresponding to the intervals in F, but our algorithm ignores the intervals in F.

We assume $\mathsf{opt}(\mathcal{I}) \geq \frac{10^3}{\lambda} \log n \cdot \log \log n$. Otherwise, since our algorithm will pick at least one interval, we already get a competitive ratio of $\frac{10^3}{\lambda} \log n \cdot \log \log n$.

We have already established that the event ξ_1 implies $D \leq 4\lceil \log_2 n \rceil$ and $|F| \leq 8\lceil \log_2 n \rceil$, and $\mathsf{Pr}(\xi_1) \geq 1 - \frac{1}{n}$. To show our result, the only ingredient missing is a lower bound on $\mathsf{opt}(\mathcal{I}_1)$. To this end, let us define ξ_2 to be the event $\mathsf{opt}(\mathcal{I}_1) \geq \frac{1-\delta}{2} \cdot \mathsf{opt}(\mathcal{I})$, where $\delta = \frac{1}{10}$. Now a similar calculation as in the proof of Lemma 11 (using Lemma 6), yields for all $n \geq 100$, $\mathsf{Pr}(\xi_2) \geq 1 - \frac{1}{n^2}$. Using union bound, the probability that both ξ_1 and ξ_2 hold is

$$\mathsf{Pr}(\xi_1 \wedge \xi_2) \geq 1 - \left(\frac{1}{n} + \frac{1}{n^2}\right) \geq 1 - \frac{2}{n}.$$

$\xi_1 \wedge \xi_2$ implies that $\mathcal{I}_1$ is a $(K = \lceil n/2 \rceil, D = 4\lceil \log_2 n \rceil)$-bounded instance, and $|F| \leq 8\lceil \log_2 n \rceil$, $\mathsf{opt}(\mathcal{I}_1) \geq \frac{9}{20}\mathsf{opt}(\mathcal{I})$. Assuming these ranges for $K, D, |F|$, and $\mathsf{opt}(\mathcal{I}_1)$, we lower bound the expression $\lambda \cdot \mathsf{opt}(\mathcal{I}_1 \setminus F)/(D + \log K \log \log K) - 1$.

$$\begin{aligned}\frac{\lambda \cdot \mathsf{opt}(\mathcal{I}_1 \setminus F)}{(D + \log K \log \log K)} - 1 &\geq \frac{\lambda \cdot \left(\frac{9}{20} \cdot \mathsf{opt}(\mathcal{I}) - 8\lceil \log_2 n \rceil\right)}{(4\lceil \log_2 n \rceil + \log\lceil \frac{n}{2} \rceil \cdot \log \log \lceil \frac{n}{2} \rceil)} - 1 \\ &\geq \frac{\lambda \cdot \frac{8}{20} \cdot \mathsf{opt}(\mathcal{I})}{10 \log n \cdot \log \log n} - 1 \geq \frac{7\lambda \cdot \mathsf{opt}(\mathcal{I})}{200 \log n \cdot \log \log n},\end{aligned}$$

where the second inequality holds since $\mathsf{opt}(\mathcal{I}) \geq 160\lceil \log_2 n \rceil$ and the last inequality holds since $\lambda \cdot \mathsf{opt}(\mathcal{I}) \geq 200 \log n \cdot \log \log n$.

Now, we show that the output of our algorithm h is at least this lower bound with high probability.

$$\begin{aligned}\mathsf{Pr}\left[h \geq \frac{7\lambda \cdot \mathsf{opt}(\mathcal{I})}{200 \log n \cdot \log \log n}\right] &\geq \mathsf{Pr}(\xi_1 \wedge \xi_2) \cdot \mathsf{Pr}\left[h \geq \frac{7\lambda \cdot \mathsf{opt}(\mathcal{I})}{200 \log n \cdot \log \log n} \mid \xi_1 \wedge \xi_2\right] \\ &\geq \left(1 - \frac{2}{n}\right)\left(1 - \frac{\mu}{\log\lceil n/2 \rceil}\right) \geq 1 - \frac{\mu + 3}{\log n}.\end{aligned}$$ ◀

The algorithm can be implemented in $O(n \log n)$ time using BST.

5.1 Extension to Hyperrectangles

We now consider the case of hyperrectangles and prove Theorem 5. As in the case of intervals, we shall have two phases, the scale preparation and outsourcing phases each comprising $n/2$ intervals each. After all hyperrectangles in the scale preparation phase have arrived, we perform non-uniform scaling in each dimension $j \in [d]$ independently. Lemma 15 then implies that the scaled hyperrectangles form a (K, D)-bounded set with $K = O(n)$ and $D = O(\log n)$, with probability at least $1 - d/n$ (by taking union bound over all dimensions). Lemma 13 then yields the promised competitive ratio guarantee in Theorem 5. The implementation details for achieving a running time of $\tilde{O}_d(n)$ are deferred to the full version [37].

6 Conclusion

We provide a guarantee of polylogarithmic strongly competitive ratio for the independent set in geometric intersection graphs for a wide range of objects. Note that we did not try to optimize the constants in the competitive ratio. With a refined analysis, the constants can be improved. Apart from being the only known algorithm with a sublinear strongly competitive ratio, due to near-linear runtime, our algorithm also provides a simple candidate algorithm to be used in practice.

References

1 Anna Adamaszek and Andreas Wiese. A quasi-PTAS for the two-dimensional geometric knapsack problem. In *SODA*, pages 1491–1505, 2015.

2 Susanne Albers, Waldo Gálvez, and Maximilian Janke. Machine covering in the random-order model. *Algorithmica*, 85(6):1560–1585, 2023.

3 Susanne Albers and Maximilian Janke. Scheduling in the random-order model. *Algorithmica*, 83(9):2803–2832, 2021.

4 Susanne Albers and Maximilian Janke. Scheduling in the secretary model. In *FSTTCS*, volume 213, pages 6:1–6:22, 2021.

5 Susanne Albers, Arindam Khan, and Leon Ladewig. Best fit bin packing with random order revisited. *Algorithmica*, 83(9):2833–2858, 2021.

6 Susanne Albers, Arindam Khan, and Leon Ladewig. Improved online algorithms for knapsack and GAP in the random order model. *Algorithmica*, 83(6):1750–1785, 2021.

7 Nikhil Ayyadevara, Rajni Dabas, Arindam Khan, and K. V. N. Sreenivas. Near-optimal algorithms for stochastic online bin packing. In *ICALP*, pages 12:1–12:20, 2022.

8 Unnar Th. Bachmann, Magnús M. Halldórsson, and Hadas Shachnai. Online selection of intervals and t-intervals. *Information and Computation*, 233:1–11, 2013.

9 Nikhil Bansal and Arindam Khan. Improved approximation algorithm for two-dimensional bin packing. In *SODA*, pages 13–25, 2014.

10 Sayan Bhattacharya, Fabrizio Grandoni, and David Wajc. Online edge coloring algorithms via the nibble method. In *SODA*, pages 2830–2842, 2021.

11 Sujoy Bhore, Jean Cardinal, John Iacono, and Grigorios Koumoutsos. Dynamic geometric independent set. *arXiv preprint arXiv:2007.08643*, 2020.

12 Sujoy Bhore, Fabian Klute, and Jelle J Oostveen. On streaming algorithms for geometric independent set and clique. In *WAOA*, pages 211–224, 2022.

13 Allan Borodin and Christodoulos Karavasilis. Any-order online interval selection. In *WAOA*, pages 175–189, 2023.

14 Joan Boyar, Lene M. Favrholdt, Shahin Kamali, and Kim S. Larsen. Online interval scheduling with predictions. In *Algorithms and Data Structures*, pages 193–207, 2023.

15 Sergio Cabello and Pablo Pérez-Lantero. Interval selection in the streaming model. *Theoretical Computer Science*, 702:77–96, 2017.

16 Parinya Chalermsook and Julia Chuzhoy. Maximum independent set of rectangles. In *SODA*, pages 892–901, 2009.

17 Timothy M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *Journal of Algorithms*, 46(2):178–189, 2003.

18 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, 2017.

19 Jana Cslovjecsek, Michal Pilipczuk, and Karol Wegrzycki. A polynomial-time opt$^{\varepsilon}$-approximation algorithm for maximum independent set of connected subgraphs in a planar graph. In *SODA*, pages 625–638, 2024.

20 Artur Czumaj, Shaofeng H-C Jiang, Robert Krauthgamer, and Pavel Veselỳ. Streaming algorithms for geometric steiner forest. In *ICALP*, pages 1–20, 2022.

21 Minati De, Saksham Jain, Sarat Varma Kallepalli, and Satyam Singh. Online piercing of geometric objects. In *FSTTCS*, 2022.

22 Minati De, Sambhav Khurana, and Satyam Singh. Online dominating set and independent set. *arXiv preprint arXiv:2111.07812*, 2021.

23 Jeffrey S Doerschler and Herbert Freeman. A rule-based system for dense-map name placement. *Communications of the ACM*, 35(1):68–79, 1992.

24 Evgenii Borisovich Dynkin. Optimal choice of the stopping moment of a Markov process. In *Doklady Akademii Nauk*, volume 150, pages 238–240. Russian Academy of Sciences, 1963.

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

26 Guy Even and Shakhar Smorodinsky. Hitting sets online and unique-max coloring. *Discrete Applied Mathematics*, 178:71–82, 2014.

27 Robert J Fowler, Michael S Paterson, and Steven L Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information processing letters*, 12(3):133–137, 1981.

28 Jacob Fox and János Pach. Computing the independence number of intersection graphs. In *SODA*, pages 1161–1165, 2011.

29 Hu Fu, Zhihao Gavin Tang, Hongxun Wu, Jinzhao Wu, and Qianfan Zhang. Random order vertex arrival contention resolution schemes for matching, with applications. In *ICALP*, 2021.

30 Takeshi Fukuda, Yasukiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. *ACM SIGMOD Record*, 25(2):13–23, 1996.

31 Stanley PY Fung, Chung Keung Poon, and Feifeng Zheng. Online interval scheduling: randomized and multiprocessor cases. *Journal of Combinatorial Optimization*, 16(3):248–262, 2008.

32 Waldo Gálvez, Fabrizio Grandoni, Afrouz Jabal Ameli, Klaus Jansen, Arindam Khan, and Malin Rau. A tight $(3/2+\varepsilon)$-approximation for skewed strip packing. *Algorithmica*, 85(10):3088–3109, 2023.

33 Waldo Gálvez, Fabrizio Grandoni, Salvatore Ingala, Sandy Heydrich, Arindam Khan, and Andreas Wiese. Approximating geometric knapsack via l-packings. *ACM Transactions on Algorithms*, 17(4):33:1–33:67, 2021.

34 Waldo Gálvez, Fabrizio Grandoni, Arindam Khan, Diego Ramírez-Romero, and Andreas Wiese. Improved approximation algorithms for 2-dimensional knapsack: Packing into multiple L-shapes, spirals, and more. In *SoCG*, volume 189, pages 39:1–39:17, 2021.

35 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 *SODA*, pages 894–905, 2022.

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

37 Mohit Garg, Debajyoti Kar, and Arindam Khan. Random-order online interval scheduling and geometric generalizations. *arXiv preprint arXiv:2402.14201*, 2024.

38 Oliver Göbel, Martin Hoefer, Thomas Kesselheim, Thomas Schleiden, and Berthold Vöcking. Online independent set beyond the worst-case: Secretaries, prophets, and periods. In *ICALP*, pages 508–519, 2014.

39 Anupam Gupta, Gregory Kehne, and Roie Levin. Random order online set cover is as easy as offline. In *FOCS*, pages 1253–1264, 2021.

40 Anupam Gupta and Sahil Singla. Random-order models. In Tim Roughgarden, editor, *Beyond the Worst-Case Analysis of Algorithms*, pages 234–258. 2020.

41 Guru Prashanth Guruganesh and Sahil Singla. Online matroid intersection: Beating half for random arrival. In *IPCO*, pages 241–253, 2017.

42 Magnús M Halldórsson, Kazuo Iwama, Shuichi Miyazaki, and Shiro Taketomi. Online independent sets. *Theoretical Computer Science*, 289(2):953–962, 2002.

43 Sariel Har-Peled. *Geometric approximation algorithms.* American Mathematical Society, 2011.

44 Anish Hebbar, Arindam Khan, and K. V. N. Sreenivas. Bin packing under random-order: Breaking the barrier of 3/2. In *SODA*, pages 4177–4219, 2024.

45 Monika Henzinger, Stefan Neumann, and Andreas Wiese. Dynamic approximate maximum independent set of intervals, hypercubes and hyperrectangles. In *SoCG*, volume 164, pages 51:1–51:14, 2020.

46 Klaus Jansen, Arindam Khan, Marvin Lira, and K. V. N. Sreenivas. A PTAS for packing hypercubes into a knapsack. In *ICALP*, volume 229, pages 78:1–78:20, 2022.

47 Claire Kenyon. *Best-fit bin-packing with random order.* PhD thesis, Laboratoire de l'informatique du parallélisme, 1995.

48 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. Primal beats dual on online packing LPs in the random-order model. *SIAM J. Comput.*, 47(5):1939–1964, 2018.

49 Arindam Khan, Aditya Lonkar, Arnab Maiti, Amatya Sharma, and Andreas Wiese. Tight approximation algorithms for two-dimensional guillotine strip packing. In *ICALP*, volume 229, pages 80:1–80:20, 2022.

50 Arindam Khan, Aditya Lonkar, Saladi Rahul, Aditya Subramanian, and Andreas Wiese. Online and dynamic algorithms for geometric set cover and hitting set. In *SoCG*, volume 258, pages 46:1–46:17, 2023.

51 Arindam Khan and Eklavya Sharma. Tight approximation algorithms for geometric bin packing with skewed items. *Algorithmica*, 85(9):2735–2778, 2023.

52 Dennis Komm, Rastislav Královic, Richard Královic, and Tobias Mömke. Randomized online computation with high probability guarantees. *Algorithmica*, 84(5):1357–1384, 2022.

53 Stefano Leonardi, Alberto Marchetti-Spaccamela, Alessio Presciutti, and Adi Rosén. On-line randomized call control revisited. *SIAM Journal on Computing*, 31(1):86–112, 2001.

54 Liane Lewin-Eytan, Joseph Seffi Naor, and Ariel Orda. Routing and admission control in networks with advance reservations. In *APPROX*, pages 215–228, 2002.

55 Richard J Lipton and Andrew Tomkins. Online interval scheduling. In *SODA*, volume 94, pages 302–311, 1994.

56 Dániel Marx. Efficient approximation schemes for geometric problems? In *ESA*, volume 3669 of *Lecture Notes in Computer Science*, pages 448–459, 2005.

57 Milena Mihail and Thorben Tröbst. Online matching with high probability. *CoRR*, abs/2112.07228, 2021.

58 Joseph SB Mitchell. Approximating maximum independent set for rectangles in the plane. In *FOCS*, pages 339–350, 2022.

59 Tobias Mömke and Andreas Wiese. Breaking the barrier of 2 for the storage allocation problem. In *ICALP*, volume 168, pages 86:1–86:19, 2020.

60 Steven S Seiden. Randomized online interval scheduling. *Operations Research Letters*, 22(4-5):171–177, 1998.

61 Gerhard J Woeginger. On-line scheduling of jobs with fixed start and end times. *Theoretical Computer Science*, 130(1):5–16, 1994.
62 Ge Yu and Sheldon H Jacobson. Primal-dual analysis for online interval scheduling problems. *Journal of Global Optimization*, 77(3):575–602, 2020.
63 Feifeng Zheng, Yongxi Cheng, Ming Liu, and Yinfeng Xu. Online interval scheduling on a single machine with finite lookahead. *Computers & operations research*, 40(1):180–191, 2013.