

Online Scheduling Intervals and t -Intervals

Unnar Th. Bachmann* Magnús M. Halldórsson*[†] Hadas Shachnai[‡]

Abstract

A t -interval is a union of at most t half-open intervals on the real line. The special case where $t = 1$ is an interval. Often, the requests for contiguous allocation of a linear resource can be modeled by a sequence of t -intervals. We consider the problems of online scheduling intervals and t -intervals, which show up in Video-on-Demand services, high speed networks and molecular biology, among others. We derive lower bounds and (almost) matching upper bounds on the competitive ratios of randomized algorithms for scheduling intervals, 2-intervals and t -intervals, for any $t > 2$. While offline t -interval scheduling has been studied before, the online version is considered here for the first time.

1 Introduction

Interval scheduling is a form of resource allocation problem, in which the machines are the resource. As argued by Kolen et al. [11], operations management has undergone a “transition in the last decennia from resource oriented logistics (where the availability of resources has dictated the planning and completion of jobs) to demand oriented logistics (where the jobs and their completion are more or less fixed and the appropriate resources must be found).” They suggest that this implies a move from traditional scheduling to *interval scheduling*.

Suppose you are running a resource online. Customers call and request to use it from time to time, for up to t time periods, not necessarily of same length. These requests must either be accepted or declined. If a request is accepted then it *occupies* the resource for these periods of time. A request cannot be accepted if one or more of its periods intersect a period of a previously accepted request. The goal is to accept as many requests as possible.

This can be modeled as the following *online t -interval scheduling (t -ISP)* problem. Let t the maximum number of periods involved in any request. Then each request is represented by a t -interval, for some $t \geq 1$, namely, a union of at most t half-open intervals on the real line. The t -intervals arrive one by one and need to be scheduled non-preemptively on a single machine. Two t -intervals, I and J , are disjoint if none of their segments intersect. They do intersect if a segment of I intersects one or more segments of J , or vice versa. Upon arrival of a t -interval, the scheduler needs to decide whether it is accepted; if not, it is lost forever. The goal is to schedule a subset of non-intersecting t -intervals of maximum cardinality. In the *weighted* version of the problem, each t -interval is associated with some positive weight that is gained if all of its segments are scheduled. The goal is to schedule a maximum weight subset of non-intersecting t -intervals. The special case where $t = 1$ is known as the *online interval scheduling problem (ISP)*. An example of an instance of online t -ISP is given in Figure 1.

*School of Computer Science, Reykjavik University, 101 Reykjavik, Iceland. {unnar07,mmh}@ru.is.

[†]Research supported by grant 060034022 from Iceland Research Foundation

[‡]Department of Computer Science, The Technion, Haifa 32000, Israel. hadas@cs.technion.ac.il.

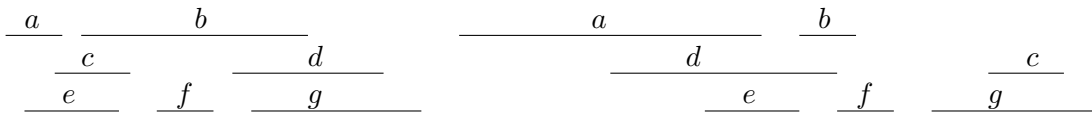


Figure 1: A linear resource is requested by customers a, b, c, d, e, f and g in that order, for two periods each. If b is accepted then each of the following requests must be declined. Thus, an optimal schedule consists of a, f and g .

The performance of an online algorithm is measured in terms of its competitive ratio. Formally, let OPT be an optimal offline algorithm for the problem. The competitive ratio of A is defined as $\sup_{\sigma} \frac{OPT(\sigma)}{A(\sigma)}$, where σ is an input sequence, and $OPT(\sigma), A(\sigma)$ are the number of t -intervals scheduled by OPT and A , respectively. For randomized algorithms, we replace $A(\sigma)$ with the expectation $\mathbb{E}[A(\sigma)]$ and define the competitive ratio as $\rho_A = \sup_{\sigma} \frac{OPT(\sigma)}{\mathbb{E}[A(\sigma)]}$. An algorithm with competitive ratio of at most ρ is called ρ -competitive. Let n be the number of intervals in the instance; also, denote by Δ the ratio between the longest and shortest intervals.

1.1 Applications

We list below several natural applications of our problems.

Crew scheduling: This is the problem of assigning flight crews to flights, where each flight has a start-time, end-time and duration. The aim is to find the minimum number of flight crews needed for a given set of flights. Each flight is represented by an interval, and each crew by a machine. The problem is to minimize the number of crews/machines needed.

Bandwidth allocation: A set of users communicate via a network with limited bandwidth. Each communication request can be thought of as an interval requiring a certain amount of bandwidth (demand). Communications requests can have different priorities. The competitiveness of the system is with respect to *throughput*, or the number of requests satisfied. Here, the online version is particularly important. Preemption usually improves the competitive ratio (see, e.g., [3]).

Video-on-Demand Service: Scheduling of continuous-media data occurs where multimedia servers broadcast streams of data to clients upon request. Requests from the clients can be modeled as t -intervals, since each request can be split into viewing-intervals and breaks.

Pattern Matching: Vialette [14] studies two problems of pattern matching over a set of 2-intervals. The first problem is to find a given 2-interval pattern and the second is to find the longest 2-interval from a given graph. This problem arises in molecular biology, where a given RNA secondary structure has to be found in a database.

Other applications include high speed networks, storage subsystems and molecular biology (see a survey in [2]).

1.2 Related Work

Scheduling intervals and t -intervals: We can view t -ISP as the problem of finding a *maximum independent set (IS)* in a t -interval graph. While for the special case of interval graphs the problem is known to be polynomially solvable (see, e.g., [10]), already for $t = 2$ the IS problem becomes APX-hard [4]. The paper [4] presents a $2t$ -approximation algorithm for the offline weighted t ISP.

Later works extended the study to scheduling t -intervals with demands, where each interval is associated with a set of segments and a demand for machine capacity [5], as well as the study of other optimization problems on t -interval graphs (see, e.g., [6]).

There is a wide literature on maximum independent set problem, in various classes of graphs. The online version of the IS problem was studied in [7], where a $\Omega(n)$ -lower bound on the competitive ratios of randomized algorithms was given, even for interval graphs (but not when interval representation is given). A survey of other works is given in [2].

Online interval scheduling: Lipton and Tomkins [12] considered an online interval scheduling problem where the intervals have weights proportional to their length and the intervals arrive by time (i.e., in order of their left endpoints). They showed that $\theta(\log \Delta)$ -competitive was optimal, when Δ is known, and introduced a technique that gives a $O(\log^{1+\epsilon} \Delta)$ -competitive factor when Δ is unknown. Woeginger [15] considered a preemptive version of weighted ISP and gave an optimal 4-competitive algorithm. Numerous results are known about interval scheduling under the objective of minimizing the number of machines, or alternatively, online coloring interval graphs. In particular, a 3-competitive algorithm was given by Kierstead and Trotter [9]. The t -ISP problem bears a resemblance to the JISP problem [13], where each job consists of several intervals and the task is to complete as many jobs as possible. The difference is that in JISP, it suffices to select only one of the possible intervals for the job.

Call admission: Similar problems have been studied also in the area of call admission. We note that ISP can be viewed as call admission on a line, where the objective is to maximize the number of accepted calls. The paper [1] presents a strongly $\lceil \log N \rceil$ -competitive algorithm for the problem, where N is the number of nodes on the line. This yields an $O(\log \Delta)$ -competitive algorithm for general ISP instances when Δ is known a-priori. We give an algorithm that achieves (almost) the same ratio for the case where Δ is unknown.

1.3 Our Results

We derive the first lower and upper bounds on the competitive ratios of online algorithms for t -ISP and new or improved bounds for ISP. Table 1 summarizes the results for various classes of instances of ISP, 2-ISP and t -ISP. Entries marked with \cdot follow by inference. All of the results apply to randomized algorithms against oblivious adversary. In comparison, proving strong lower bounds for deterministic algorithms (including a lower bound of Δ for ISP) is straightforward. The upper bounds for general inputs are for the case where Δ , the ratio between the longest and shortest segment in the instance, is unknown in advance. The lower bound of $\Omega(\log \Delta)$ for ISP follows from the results of [1].

	ISP		2-ISP		t -ISP	
	<i>u.b.</i>	<i>l.b.</i>	<i>u.b.</i>	<i>l.b.</i>	<i>u.b.</i>	<i>l.b.</i>
General inputs	$O(\log^{1+\epsilon} \Delta)$	$\Omega(\log \Delta)$	$O(\log^{2+\epsilon} \Delta)$	$\Omega(\log \Delta)$	—	\cdot
Two lengths	4	4	8	6	—	\cdot
Unit length	2	2	4	3	$O(t)$	$\Omega(t)$
Bounded depth	$3/2$ ($s = 2$)	$2 - 1/s$	—	—	—	—

Table 1: Results for online scheduling intervals and t -intervals

Contribution: In Section 2 we present a *stacking technique* that we use throughout the paper to derive lower bounds for randomized algorithms. Using the linearity of the representation of interval graphs as segments on the line, we extend the construction to obtain randomized lower bounds for t -interval graphs, for any $t > 1$.

Due to space constraints, some of the results are omitted and some are relegated to the Appendix. The detailed results will be given in the full version of the paper (see also in [2]).

2 Technique: Stacking Construction

In our study of randomized algorithms for (t -)interval scheduling, we use the following technique. When deriving lower bound for deterministic algorithms, the adversary takes advantage of the fact that it can foresee the outcome of a deterministic algorithm. He “stacks” the intervals so as to achieve a desirably poor outcome by the algorithm. The general idea is similar to a lower bounding technique of Awerbuch et al [1] for call control. We describe the technique below.

Let R be an ISP-algorithm and let parameters q and x be given. A (q, x) -*stacking construction* for R is formed as follows. Form q unit intervals I_1, \dots, I_q that mutually overlap with left endpoints spaced x/q apart towards the left. Namely, $I_i = [x(1-i/q), 1+x(1-i/q))$, for $i = 1, \dots, q$. Let p_i be the probability that R schedules I_i . The adversary knows the values p_i and forms its construction accordingly. Namely, let m be the smallest value such that $p_m \leq 1/q$. Since $\sum_i p_i \leq 1$, there must be at least one such value. The input sequence construction consists of $\langle I_1, I_2, \dots, I_m, J_m \rangle$, where $J_m = [x(1-m/q), 1+x(1-m/q))$. This is illustrated in Fig. 2.

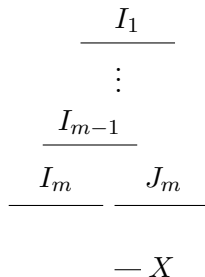


Figure 2: (q, x) -stacking construction.

Lemma 1 *A (q, x) -stacking construction \mathcal{I} has the following properties.*

1. *All intervals in \mathcal{I} overlap the segment $[1, 1+x)$.*
2. *All intervals in \mathcal{I} are contained within the interval $[0, 2+x)$.*
3. *The intervals in $\mathcal{I} \setminus \{I_m\}$ have a common intersection of length x/q , given by the segment $X = I_{m-1} \cap J_m = [1+x(1-m/q), 1+x(1-(m-1)/q))$.*
4. *$\mathbb{E}_R[I_m] = p_m \leq 1/q$. Thus, $\mathbb{E}_R[\mathcal{I}] \leq 1 + 1/q$,*
5. *$OPT(\mathcal{I}) = 2$. Thus, the performance ratio of R is at least $2/(1 + 1/q)$.*

By taking q arbitrarily large, we obtain the following performance bound.

Theorem 2 *Any randomized online algorithm for ISP with unit intervals has competitive ratio at least 2.*

We can imitate the stacking construction with 2-intervals by repeating the construction for both segments. We refer to this as a *2-interval (q, x) -stacking construction*.

We shall also use the stacking construction *shifted* by a displacement f , by adding f to the starting point of each interval. We may also use intervals of non-unit length.

3 Online Interval Scheduling

3.1 Unit Intervals and Depth

We give upper and lower bounds on the competitive ratios for ISP with unit intervals. We parameterize the problem in terms of the *depth* of the interval system, which is the maximum number of intervals that overlap a common point. This corresponds to the clique number of the corresponding interval graph.

Theorem 3 *The competitive ratio of any randomized algorithm for ISP of unit intervals is at least $2 - 1/s$, where s is the depth of the instance.*

Proof. We modify the $(s, 1)$ -stacking construction slightly. Let p_i be the probability that the given algorithm R selects interval I_i . If $p_1 \leq 1/(2 - 1/s) = s/(2s - 1)$, then we stop with the unit sequence $\langle I_1 \rangle$. The performance ratio is then at least $1/p_1 \geq 2 - 1/s$. Otherwise we stop the sequence at I_m , where m is the smallest number such that $p_m \leq 1/(2s - 1)$. This is well defined since $s/(2s - 1) + \sum_{i=2}^s 1/(2s - 1) = 1$. As before, this is followed by the interval J_m intersecting only the first $m - 1$ intervals. The algorithm obtains expected value at most $1 + p_m \leq 1 + 1/(2s - 1) = 2s/(2s - 1)$, versus 2 for the optimal solution. The above procedure can be repeated arbitrarily often, ensuring that the lower bound holds also in the asymptotic case. ■

We now describe a randomized algorithm that achieves the above ratio for $s = 2$. Consider algorithm, `Random_or_Greedy` (**RoG**), which handles an arriving interval as follows. If the interval does not overlap any previously presented interval, schedule it with probability $2/3$, else schedule the interval greedily.

Theorem 4 *Algorithm `RoG` is $3/2$ -competitive for unit intervals with depth 2.*

Proof. Assume that the instance is connected; otherwise, we can argue the bound for each component separately.

The depth restriction means that each interval can intersect at most two other intervals: one from the left and one from the right. The instance is therefore a chain of unit intervals. We divide the intervals into three types, based on the number of previous intervals the given interval intersects. A type- i interval, for $i = 0, 1, 2$, intersects i previously presented intervals. Two type-2 intervals cannot intersect, as otherwise the one that appears earlier will have degree 3, leading to depth at least 3. The instance consists therefore of chains of type-0 and type-1 intervals attached together by type-2 intervals. Each chain is started by a type-0 interval, followed by type-1 intervals. It follows that, if n_i denotes the number of intervals of type i , we have that

$$n_0 \geq n_2 + 1 . \tag{1}$$

Consider now the unconditional probability that intervals of each type are selected, i.e. the probability independent of other selections. The probability of type-0 intervals being selected is $2/3$. The probability of the selection of type-1 intervals alternates between $1/3$ and $2/3$. It suffices for our claim to know that the probability of selecting type-2 intervals is positive.

The expected number of intervals selected by the algorithm is then, using (1), bounded below by

$$\frac{2}{3}n_0 + \frac{1}{3}n_1 \geq \frac{1}{3}(n_0 + n_1 + n_2 + 1) = \frac{n+1}{3}.$$

On the other hand, the number of intervals in an optimal schedule is the independence number of the n -path, or $\lceil \frac{n}{2} \rceil \leq \frac{n+1}{2}$. Hence, the competitive ratio is at most $3/2$. ■

3.2 ISP with intervals of two lengths

Consider now ISP instances where the intervals can be of two different lengths, 1 and d . It is easy to argue a 4-competitive algorithm by the classic Classify-and-Select approach: Flip a coin, choosing either the unit intervals or the length- d intervals, and then greedily adding intervals of that length only.

We find that it is not possible to significantly improve on that very simple approach; the proof is given in the appendix.

Theorem 5 *Any randomized online algorithm for ISP with intervals of two lengths 1 and d has performance ratio at least 4, asymptotically with d .*

3.3 ISP with Parameter n

ISP is easily seen to be difficult on instances without constraints on the size of the intervals. The adversary keeps introducing disjoint intervals until the algorithm selects one of them, I ; the remaining intervals presented will then be contained in I . This leaves the algorithm with a single interval, while the optimal solution contains the rest, for a ratio of $n - 1$.

It is less obvious that a linear lower bound holds also for randomized algorithms against oblivious adversary.

Theorem 6 *Any randomized online algorithm for ISP has competitive ratio $\Omega(n)$.*

Proof. Let $n > 1$ be an integer. Let r_1, r_2, \dots, r_{n-1} be a sequence of uniformly random bits. Let the sequence x_1, x_2, \dots, x_n of points be defined inductively by $x_1 = 0$ and $x_{i+1} = x_i + r_i \cdot 2^{n-i}$. We construct a sequence \mathcal{I}_n of n intervals I_0, I_1, \dots, I_n , where $I_i = [x_i, x_i + 2^{n-i})$, for $i = 1, \dots, n$.

The collection $A = \{I_i : r_i = 1\} \cup \{I_n\}$ forms an independent set, informally referred to as the “good” intervals. The set $B = \mathcal{I}_n \setminus A = \{I_i : r_i = 0\}$ forms a clique; informally, these are the “bad” intervals.

Consider a randomized algorithm A and the sequence of intervals chosen by A . The event that a chosen interval is good is a Bernoulli trial, and these events are independent. Thus, the number of intervals chosen until a bad one is chosen is a geometric random variable with a mean 2. Even accounting for the last interval, which is known to be good, the expected number of accepted intervals $\mathbb{E}[\sigma_R]$ is at most 3.

On the other hand, the expected number of good intervals is $(n-1)/2 + 1$, and so the expected size of the optimal solution is $n/2$. By standard arguments, this holds also with high probability, up to lower order terms. The competitive ratio of R on \mathcal{I}_n is therefore at least $n/6$. ■

Notice that in Theorem 6, the intervals are presented in order of increasing endpoints. Thus, the bound holds also for the scheduling-by-time model. The adversary in Theorem 6 has also the property of being *transparent* [8] in the sense that as soon as the algorithm has made its decision on an interval, the adversary reveals his own choice.

4 Online 2-Interval Scheduling

4.1 Unit Segments

Theorem 7 *Any randomized online algorithm for 2-ISP of unit intervals has competitive ratio at least 3.*

Proof. Consider any randomized online 2-ISP algorithm R . Let q be an even number and let $q' = 3q/2$.

We start with 2-interval $(q', 1)$ -stacking construction \mathcal{I} for R . Recall that the expected gain of R on interval I_m is $\mathbb{E}_R[I_m] \leq 1/q'$. Let p be the probability that R selects some interval in $\mathcal{T}' = \mathcal{I} \setminus \{I_m\}$. If $p < 2/3$, then we stop the construction. The expected solution size found by R is then $\mathbb{E}_R[\mathcal{I}] \leq p + 1/q'$, while the optimal solution is of size 2, for a ratio of $2/(p + 1/q') \geq 2/(2/3 + 2/(3q)) = 3/(1 + 1/q)$.

Assume therefore that $p \geq 2/3$. Let X_1 be the common intersection of the first segments of the 2-intervals in \mathcal{T}' , and X_2 be the common intersection of the second segments. Let f_i denote the starting point of X_i , $i = 1, 2$. By Lemma 1, the length of each X_i is $1/q'$.

We now form a (q, x) -stacking construction \mathcal{I}_2 of 2-intervals for R shifted by f_1 , where $x = |X_1| = 1/q'$. Thus, the first segments are positioned to overlap X_1 , where $x = |X_1| = 1/q'$; the second segments are immaterial as long as they don't intersect any previous intervals. We then do an identical construction \mathcal{I}_3 shifted by f_2 ; again, the second segments do not factor in. This completes the construction.

We can make the following observations about the combined construction $\mathcal{J} = \mathcal{I} \cup \mathcal{I}_2 \cup \mathcal{I}_3$:

Observation 4.1 1. All intervals in \mathcal{I}_2 overlap X_1 .

2. All intervals in \mathcal{I}_3 overlap X_2 .

3. $OPT(\mathcal{J}) = 4$, given by $I_m^2, J_m^2, I_m^3, I_m^3$.

4. $\mathbb{E}_R[\mathcal{I}_2] \leq (1 - p)(1 + 1/q)$, by Lemma 1 (3) and part 1 above.

It follows that

$$\mathbb{E}_R[\mathcal{J}] = \mathbb{E}_R[I_m] + \mathbb{E}_R[\mathcal{T}'] + \mathbb{E}_R[\mathcal{I}_2] + \mathbb{E}_R[\mathcal{I}_3] \leq 1/q' + p + 2(1 - p)(1 + 1/q) = 2 - p + (4/3 - 2p)q .$$

Since $p \geq 2/3$, $\mathbb{E}_R[\mathcal{J}] \geq 2 - p$ and the performance ratio of R on \mathcal{J} is $OPT(\mathcal{J})/\mathbb{E}_R[\mathcal{J}] \geq 4/(4/3) = 3$.

■

4.2 Segments of Two Lengths

In this section we give an 8-competitive algorithm for 2-ISP where the 2-intervals can have lengths 1 and $d \gg 1$. A lower bound of 6 is omitted in this version.

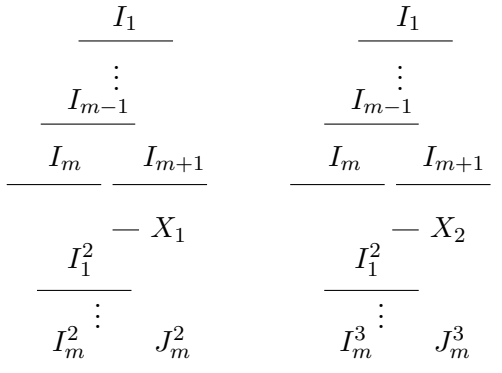


Figure 3: Construction of a lower bound of 3 for unit 2-ISP

Given an instance S of 2-ISP, consider the following algorithm, A_v , which can either schedule a 2-interval, reject it, or schedule it *virtually*.¹ A virtually scheduled interval does not occupy the resource; however, it blocks other 2-intervals from being scheduled. Assume that the length of each segment is either 1 (*short*) or d (*long*). In scheduling a 2-interval I , A_v applies the following rules, which depend on the availability of the resource.

1. **The resource is free.** Schedule I greedily if both of its segments are short; otherwise, schedule I with probability $1/2$ and virtually schedule it with probability $1/2$.
2. **I intersects a virtually scheduled 2-interval J .** If I contains two short segments then schedule I greedily, else if there is no intersection between a long segment of I and a long segment of J then schedule I with probability $1/2$.

Our analysis of A_v uses the following charging scheme. Let S_{OPT} be the subset of 2-intervals selected by an optimal offline algorithm, OPT , then for any $J \in S_{\text{OPT}}$, we assign $w(J, J) = 1$; for $I \in S \setminus S_{\text{OPT}}$ such that $J \cap I \neq \emptyset$, we assign a weight $w(I, J) \in [0, 1]$. Intuitively, if I is scheduled (thus blocking J) we credit J by the amount $w(I, J)$. The weights should be assigned such that, for any $I \in S$, $\sum_{J \in S_{\text{OPT}}} w(I, J) \leq 1$. Given an online algorithm, A , for any $J \in S_{\text{OPT}}$, let $w(\text{bucket}(J)) = \sum_{I \in S_A} w(I, J)$, where S_A is the subset of 2-intervals selected by A . Then to show that algorithm A is c -competitive it suffices to prove that, for any $J \in S_{\text{OPT}}$, $w(\text{bucket}(J)) \geq 1/c$. If A is randomized, we need to show that $\mathbb{E}[w(\text{bucket}(J))] \geq 1/c$.

Consider intervals $J \in S_{\text{OPT}}$ and $I \notin S_{\text{OPT}}$ that overlap. We say that J is a *terminal 2-interval* of I if a long segment of I intersects an endpoint of a short segment of J . We assign the weights $w(I, J)$ as follows.

1. If segments of the same length overlap, or if a short segment of I overlaps a long segment of J then $w(I, J) = 1/4$.
2. If J is a terminal 2-interval of I then $w(I, J) = 1/4$.
3. If I overlaps a short segment of J with a long segment, but J is not a terminal 2-interval of I , then $w(I, J) = 0$.

¹The term was used before, e.g., in [12].

From the above we have that, for all $I \notin S_{\text{OPT}}$, $\sum_{J \in S_{\text{OPT}}} w(I, J) \leq w(I) = 1$.

Theorem 8 *Algorithm A_v is 8-competitive for online 2-ISP with segments of length 1 and d .*

Proof. Given an instance S of 2-ISP with segments of length 1 and d , we use the above weight. In deriving a lower bound of $1/8$ for $\mathbb{E}[w(\text{bucket}(J))]$, for all $J \in S_{\text{OPT}}$, we distinguish between five cases.

1. The resource is free when J is presented. Then J is scheduled with probability at least $1/2$. Therefore, $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2$.
2. A 2-interval I is scheduled, or virtually scheduled prior to arrival of J , and they intersect with segments of same length. Then $w(I, J) = 1/4$, and $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot 1/4 = 1/8$.
3. A 2-interval I is presented prior to the arrival of J and intersects a long segment of J with a short segment. Then $w(I, J) = 1/4$. Since I is scheduled with probability $1/2$, $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot 1/4 = 1/8$.
4. A 2-interval I is presented prior to the arrival of J and intersects a short segment of J with a long segment. If J is a terminal 2-interval of I then $w(I, J) = 1/4$. In this case $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot 1/4 = 1/8$.

If, on the other hand, J is not a terminal 2-interval of I , then $w(I, J) = 0$. We distinguish between three scenarios.

- a) No 2-interval is presented in between I and J , or 2-intervals which intersect a long segment of J with a long segment. In this case, every 2-interval that comes in between J and I is blocked by I . Therefore, $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot 1/2 = 1/4$.
- b) A 2-interval, I' , with two short segments is presented in between I and J and overlaps the short segment of J covered by I . Since $w(I', J) = 1/4$ and I' is scheduled greedily, $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot 1/4 = 1/8$.
- c) For some $m > 1$, a set of m 2-intervals with one segment long and the other short are presented in between I and J ; all of these 2-intervals intersect J with a short segment. Then the probability that J is scheduled is $1/2 \cdot (1/2)^{m+1}$, since I , as well as the m 2-intervals presented between I and J , are scheduled virtually with probability $1/2$. Since the assigned weight of all of these intervals is $1/4$, the expected value of their total assigned weight is $1/2 \cdot \sum_{i=1}^m (1/2)^i \cdot 1/4$.

Thus, $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot (\sum_{i=1}^m (1/2)^i \cdot 1/4 + (1/2)^{m+1}) \geq 1/8$, for all $m \in \mathbb{N}$.

5. Two 2-intervals I' and I'' are presented before J , which has two short segments, and I' and I'' cover both segments of J with long segments. Assume w.l.o.g. that I'' is presented later than I' .
 - a) No more intervals in between I'' and J are presented, or 2-intervals that are blocked. In this case, $\mathbb{E}[w(\text{bucket}(J))] = 1/2 \cdot 1/2 \cdot 1 = 1/4$, since I' and I'' are scheduled virtually with probability $1/2$, and J is scheduled greedily.
 - b) Two different 2-intervals are presented and overlap both segments of J with a short segment. The assigned weight of both of them to J is $1/4$, and both are scheduled with probability at least $1/2$. Thus, $\mathbb{E}[w(\text{bucket}(J))] \geq 1/2 \cdot 1/2 \cdot 1/4 + 1/2 \cdot 1/2 \cdot 1/4 = 1/8$.

- c) Greedily scheduled interval is presented in between I'' and J and overlaps both segments of J . In this case, the assigned weight to J is $1/2$ ($1/4$ from each segment). Then $\mathbb{E}[w(\text{bucket}(J))] = 1/2 \cdot 1/2 \cdot 1/2 = 1/8$.
- d) All possible 2-intervals (not blocked) that are presented in between I'' and J intersect one segment of J , as in 4c). For simplicity assume that this segment is covered by I' . The only influence I'' has on the expected weight of $\text{bucket}(J)$ is to make sure J is scheduled with probability at least $1/2$. This case is therefore identical to 4c).

By 1.-5., $\mathbb{E}_{A_v}[w(\text{bucket}(J))] \geq 1/8$. ■

4.3 Segments of Arbitrary Lengths

Consider now more general instances of 2-ISP, in which the ratio between the longest and shortest segment is Δ , for some $\Delta > 1$. W.l.o.g. we may assume that the short segment is of length 1. We partition the set of first segments to $K = \lceil \log \Delta \rceil$ groups, such that the segments in group i have lengths in $[2^{i-1}, 2^i)$, $1 \leq i \leq K$. Similarly, we partition to K groups the second segments. A 2-interval whose first segment is of length in $[2^{i-1}, 2^i)$, and whose second segment is of length $[2^{j-1}, 2^j)$, $1 \leq i, j \leq K$, is in group (i, j) .

We now apply algorithm A_v to 2-ISP instances where the *short* segment can have a length in $[1, 2)$, and the *long* segment has length in $[d, 2d)$.

A_v makes scheduling decisions as before, using the new definitions of ‘short’ and ‘long’ segments. The proof of the next result is given in the Appendix.

Theorem 9 *Algorithm A_v is 12-competitive for 2-ISP instances with segments of two types: short with lengths in $[1, 2)$, and long with lengths in $[d, 2d)$.*

Now, given a general instance of 2-ISP, suppose that Δ is known a-priori. Consider algorithm A_{vg} which applies A_v on groups of 2-intervals. The instance is partitioned to $K^2 = \lceil \log \Delta \rceil^2$ groups, depending on the lengths of the first and second segments of each 2-interval. A_{vg} selects uniformly at random a group (i, j) , $1 \leq i, j \leq K$ and considers scheduling only 2-intervals in this groups. All other 2-intervals are declined. The next result follows from Theorem 9.

Theorem 10 *A_{vg} is $O(\log^2 \Delta)$ -competitive for 2-ISP with intervals of various lengths, where Δ is known in advance.*

For the case where Δ is *unknown* a-priori, consider algorithm \tilde{A}_{vg} , which proceeds as follows. A presented 2-interval I is in the same group as a previously presented 2-interval I' , if the ratio between the length of the first/second segment of I and I' is between 1 and 2. If not, I belongs to a new group.

We can keep track of the maximum length of a segment seen so far, ℓ and schedule 2-intervals in group i with probability

$$c_i = \frac{1}{\zeta(1 + \epsilon)((\log \ell)^2)^{1 + \epsilon/2}} = \frac{1}{\zeta(1 + \epsilon)(\log \ell)^{2 + \epsilon}},$$

where

$$\zeta(x) = \sum_{i=1}^{\infty} \frac{1}{i^x} < \infty, \text{ if } x > 1,$$

is the *Riemann zeta function*.

Theorem 11 \tilde{A}_{vg} is $O(\log^{2+\epsilon} \Delta)$ -competitive for 2-ISP with intervals of various lengths, where Δ is unknown in advance.

Proof. \tilde{A}_{vg} modifies ℓ at most $\log^2 \Delta$ times. Let S_{ij} denote the set of 2-intervals in group (i, j) . Then we get a probability distribution on S_{ij} :

$$\sum_{i=1}^{(\log \Delta)^2} c_i \leq \sum_{i=1}^{\infty} \frac{1}{\zeta(1 + \epsilon/2) i^{1+\epsilon/2}} = \frac{1}{\zeta(1 + \epsilon/2)} \sum_{i=1}^{\infty} \frac{1}{i^{1+\epsilon/2}} = \zeta(1 + \epsilon/2) \cdot \frac{1}{\zeta(1 + \epsilon/2)} = 1.$$

A 2-interval presented to \tilde{A}_{vg} is either in a new group, a selected group or a rejected one. If it is a new group and no group has been selected then either its group is chosen with probability p_i or not with probability $1 - p_i$.

Algorithm \tilde{A}_{vg} uses the probability c_i to find out the probability p_i with which to choose the i th group when a 2-interval from this group is presented. Before a 2-interval in group i arrives the algorithm has to refuse to schedule 2-intervals from certain number of groups. Therefore, the algorithm needs to ensure that $c_i = p_i \cdot \prod_{j=1}^{i-1} (1 - p_j)$. Then $p_1 = c_1$, and $p_i = \frac{c_i}{\prod_{j=1}^{i-1} (1 - p_j)}$. We note that $0 < p_i < 1$ always holds. This can be shown by induction. For $i = 1$ we have that $p_1 = c_1 < 1$. Assume that $0 < p_i < 1$ for $i = 1, \dots, k - 1$, then we get that

$$\begin{aligned} 0 < p_k &= \frac{c_k}{\prod_{j=1}^{k-1} (1 - p_j)} = \frac{c_k}{(1 - p_{k-1}) \cdot \frac{c_{k-1}}{p_{k-1}}} \\ &= \frac{\frac{1}{\zeta(1+\epsilon/2)k^{1+\epsilon/2}}}{\frac{1}{\zeta(1+\epsilon/2)(k-1)^{1+\epsilon/2}}} \cdot \frac{p_{k-1}}{1 - p_{k-1}} < \frac{p_{k-1}}{1 - p_{k-1}} \leq \frac{\frac{1}{2}}{1 - \frac{1}{2}} = 1, \end{aligned}$$

since the function $f(x) = \frac{x}{1-x}$ has an absolute extreme in $x = \frac{1}{2}$, on the interval $(0, 1]$.

The probability that \tilde{A}_{vg} chooses a single group is at least $c_{\log^{2+\epsilon} \Delta}$. After selecting a group as above, \tilde{A}_{vg} uses A_{vg} to schedule the 2-intervals in the selected group. For a given group, S_{ij} , we have:

$$\mathbb{E}[\tilde{A}_{vg}(S_{ij})] \geq c_{\log^{2+\epsilon} \Delta} \cdot \mathbb{E}[A_{vg}(S_{ij})] \geq \frac{1}{(\log \Delta)^{2+\epsilon}} \cdot \frac{1}{12} \cdot \mathbb{E}[OPT(S_{ij})].$$

By (2), \tilde{A}_{vg} is $O(\log^{2+\epsilon} \Delta)$ -competitive. ■

5 Online t -Interval Scheduling

We show here that any online algorithm for t -ISP is $\Omega(t)$ -competitive. This is done by a *reduction* to a known problem; this is standard for offline problems but rather unusual approach in the online case. We reduce the problem to the online version of the independent set (IS) problem in graphs: given vertices one by one, along with edges to previous vertices, determine for each vertex whether to add it to a set of independent vertices.

Theorem 12 Any randomized online algorithm for t -ISP with unit segments has competitive ratio $\Omega(t)$.

Proof. Let n be a number. We show that any graph on n vertices presented vertex by vertex can be converted on-the-fly to an n -interval representation. Then, an $f(t)$ -competitive online algorithm for t -ISP applied to the n -interval representation yields an $f(n)$ -competitive algorithm for the independent set problem. As shown in [7] (and follows also from Theorem 6), there is no cn -competitive algorithm for the online IS problem, for some fixed $c > 0$. The theorem then follows.

Let $G = (V, E)$ be a graph on n vertices with vertex sequence $\langle v_1, v_2, \dots, v_n \rangle$. Given vertex v_k and the induced subgraph $G[\langle v_1, v_2, \dots, v_i \rangle]$, form the n -interval I_i by

$$I_i = \bigcup_{j=1}^i X_{ij}, \quad \text{where} \quad X_{ij} = \begin{cases} [nj + i, nj + i + 1) & \text{if } j < i \text{ and } (i, j) \in E \\ [ni + j, ni + j + 1) & \text{otherwise.} \end{cases}$$

Observe that $I_i \cap I_j \neq \emptyset$ iff $(i, j) \in E$. Hence, solutions to the t -ISP instance are in one-one correspondence with independent sets in G . ■

A greedy selection of t -intervals yields a $2t$ -competitive algorithm for unit t -ISP, implying that the bound above is tight.

References

- [1] B. Awerbuch, Y. Bartal, A. Fiat, and A. Rosén. Competitive non-preemptive call control. In *SODA '94: Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 312–320, Philadelphia, PA, USA, 1994. Society for Industrial and Applied Mathematics.
- [2] U. T. Bachmann. *Online t -Interval Scheduling*. MSc thesis, School of CS, Reykjavik Univ., Dec. 2009.
- [3] A. Bar-Noy, R. Canetti, S. Kutten, Y. Mansour, and B. Schieber. Bandwidth allocation with preemption. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 616–625, New York, NY, USA, 1995. ACM.
- [4] R. Bar-Yehuda, M. M. Halldórsson, J. S. Naor, H. Shachnai, and I. Shapira. Scheduling split intervals. In *SODA '02: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 732–741, Philadelphia, PA, USA, 2002. Society for Industrial and Applied Mathematics.
- [5] R. Bar-Yehuda and D. Rawitz. Using fractional dual to schedule split intervals with demands. *Lecture Notes in Computer Science*, 3669/2005:714–725, 2005.
- [6] A. Butman, D. Hermelin, M. Lewenstein, and D. Rawitz. Optimization problems in multiple-interval graphs. In *SODA*. ACM-SIAM, 2007.
- [7] M. M. Halldórsson, K. Iwama, S. Miyazaki, and S. Taketomi. Online independent sets. *Theoretical Computer Science*, 289(2):953 – 962, 2002.
- [8] M. M. Halldórsson and M. Szegedy. Lower bounds for on-line graph coloring. *Theoretical Comput. Sci.*, 130:163–174, Aug. 1994.
- [9] H. A. Kierstead and W. T. Trotter. An extremal problem in recursive combinatorics. In *Proc. 12th Southeastern Conf. on Combinatorics, Graph Theory, and Computing. Congressus Numerantium XXXIII*, pages 143–153, 1981.

- [10] J. Kleinberg and E. Tardos. *Algorithm Design*. Addison Wesley, 2005.
- [11] A. W. Kolen, J. K. Lenstra, C. H. Papadimitriou, and F. C. Spieksma. Interval scheduling: A survey. *Naval Research Logistics*, 54:530–543, 2007.
- [12] R. J. Lipton and A. Tomkins. Online interval scheduling. In *SODA '94 Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 302–311, 1994.
- [13] F. Spieksma. On the approximability of an interval scheduling problem. *J. Sched.*, 2:215–227, 1999.
- [14] S. Vialette. On the computational complexity of 2-interval pattern matching problems. *Theor. Comput. Sci.*, 312(2-3):223–249, 2004.
- [15] G. J. Woeginger. On-line scheduling of jobs with fixed start and end times. *Theor. Comput. Sci.*, 130(1):5–16, 1994.

A Some Proofs

Proof of Theorem 5: Consider any randomized online ISP algorithm R . Let d be given and let D be the largest square number satisfying $D \leq d$. Let $q = \sqrt{D}$.

We start with a (q, d) -stacking construction \mathcal{I} for R using intervals of length d . Recall that the expected gain of R on interval I_m is $\mathbb{E}_R[I_m] \leq 1/q$. Let p be the probability that R selects some interval in $\mathcal{I}' = \mathcal{I} \setminus \{I_m\}$. If $p < 1/2$, then we stop the construction. The expected solution size found by R is then $\mathbb{E}_R[\mathcal{I}] \leq p + 1/q$, while the optimal solution is of size 2, for a ratio of $2/(p + 1/q) \geq 2/(1/2 + 1/q) = 4/(1 + 2/q)$.

Assume therefore that $p \geq 1/2$. Let X be the common intersection of interval in \mathcal{I}' and let f denote the starting point of X . By Lemma 1, the length of each X is $d/q \geq \sqrt{d}$. Let $s = q/3$. We now form a sequence of s disjoint $(q, 1)$ -stacking constructions of unit intervals, all contained within the span of X . Since the width of each stacking construction is at most 3, these constructions can all fit. Let $\hat{\mathcal{I}}$ denote the union of these s gadgets and let $\mathcal{J} = \mathcal{I} \cup \hat{\mathcal{I}}$. This completes the construction.

Observe that $OPT(\mathcal{J}) = 2s$. All intervals in $\hat{\mathcal{I}}$ overlap X . The expected gain of the algorithm on \mathcal{J} is

$$\mathbb{E}_R[\mathcal{J}] \leq \mathbb{E}_R[I_m] + (1 - p)\mathbb{E}_R[\hat{\mathcal{I}}] \leq 1/q + (1 - p)s(1 + 1/q) = (1 - p)(q/3 + 1/3) + 1/q .$$

Recall that $p \geq 1/2$. Then, the performance ratio of R on \mathcal{J} is at least

$$\frac{OPT(\mathcal{J})}{\mathbb{E}_R[\mathcal{J}]} \geq \frac{2s}{s/2(1 + 1/q) + 1/q} = \frac{4}{1 + 1/q + 1/(6q^2)} .$$

Since $q = \theta(\sqrt{d})$, this approaches 4 as d goes to infinity. ■

Proof of Theorem 9: Let S be an instance of 2-ISP with short and long segments, such that the lengths of short (long) segments can be within factor of 2 of each other. We use the following assignment of weights to 2-intervals. Let $J \in S_{OPT}$ and $I \notin S_{OPT}$.

1. $w(I, J) = 1/6$ if I overlaps J , and segments from the same length group overlap, or if a short segment of I overlaps a long segment of J .

2. J is a *terminal 2-interval* of I if a long segment of I overlaps an endpoint of a short segment of J . If J is a terminal 2-interval of I , then $w(I, J) = 1/6$.
3. $w(I, J) = 0$ if J overlaps a short segment of J with a long segment, and J is not a terminal 2-interval of I .

Using the above weights, the proof is similar to the proof of Theorem 8 (we omit the details). ■