Online Selection of 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. An interval is the special case where $t = 1$. Requests for contiguous allocation of a linear resource can be modeled as a sequence of t-intervals. We consider the problems of online selection of 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 selecting intervals, 2-intervals and t-intervals, for any $t > 2$. While offline t-interval selection has been studied before, the online version is considered here for the first time.

1 Introduction

Interval scheduling is a form of a resource allocation problem, in which the machines are the resource. As argued by Kolen et al. [12], 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 selection (t-ISP)* problem. Let t be the maximum number of periods involved in any request. Each request is represented by a t-interval, namely, a union of at most t half-open intervals (referred to as *segments*) 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, and intersect if a segment of one intersects a segment of the other. 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 select a subset (or "form a schedule") of non-intersecting t-intervals of maximum cardinality. The special case where $t = 1$ is known as the *online interval selection problem (*ISP*)*. An example of an instance of online t-ISP is given in Figure 1.

*Supported by Icelandic Research Fund (grant 060034022)

†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.

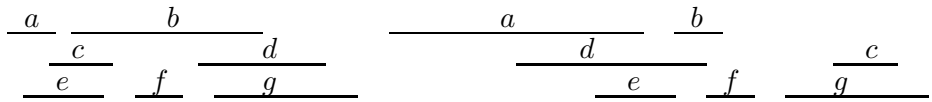

Figure 1: A linear resource is requested by customers a, b, c, d, e, f and g in that order, for two periods each. If b is accepted then each of the following requests must be declined. An optimal selection consists of a, f and g.

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 selected 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 segment lengths.

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., [4]).

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 [15] 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

Selecting 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., [11]), already for $t = 2$ the IS problem becomes APX-hard [5]. The paper [5] presents a $2t$-approximation algorithm for the offline weighted tISP. Later works extended the study to the selection of t-intervals with demands, where each interval is

associated with a set of segments and a demand for machine capacity [6], as well as the study of other optimization problems on t-interval graphs (see, e.g., [7]).

There is a wide literature on the maximum independent set problem in various classes of graphs. The online version of the IS problem was studied in [8], where a $\Omega(n)$-lower bound on the competitive ratios of randomized algorithms was given, even for interval graphs (but not when the interval representation is given). A survey of other works is given in [2].

Online interval selection: Lipton and Tomkins [13] considered an online interval selection 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 factor 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 [16] 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 [10]. The t-ISP problem bears a resemblance to the JISP problem [14], 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 segments of 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. 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 $\Delta + 1$ for ISP) is straightforward. The upper bounds for general inputs are for the case where Δ is unknown in advance.

	ISP		2-ISP		t-ISP	
	u.b.	*l.b.*	*u.b.*	*l.b.*	*u.b.*	*l.b.*
General inputs	$O(\log^{1+\varepsilon} \Delta)$	$\Omega(\log \Delta)$†	$O(\log^{2+\varepsilon} \Delta)$	$\Omega(\log \Delta)$†	–	·
Two lengths	4	4	16	6	–	·
Unit length	2 †	2	4 †	3	$2t$ †	$\Omega(t)$
Bounded depth s	$3/2$ $(s = 2)$	$2 - 1/s$	–	–	–	–

Table 1: Results for randomized online interval and t-interval selection. Entries marked with · follow by inference. Entries marked with † were known; the lower bound for ISP follows from [1], while the upper bounds for unit lengths are trivial.

Due to space constraints, some of the results (or proofs) are omitted. The detailed results

appear in [3] (see also [2]).

2 Technique: Stacking Construction

When deriving lower bounds for randomized algorithms for (t-)interval selection, we use the following technique. The adversary can take advantage of the fact that he can foresee the probability with which the algorithm selects any given action, even if he doesn't know the outcome. He presents intervals on top of each others, or "stacks" them, until some interval is chosen with sufficiently low probability. The adversary uses that to force a desirably poor outcome for the algorithm. The general idea is similar to a lower bounding technique of Awerbuch et al. [1] for call control.

Let R be an Isp-algorithm and let parameters q and x be given. A (q, x)-*stacking construction* for R is a collection of intervals formed as follows, where q is an upper bound on the number of intervals stacked and x is the extent to which intervals can be shifted. Form q unit intervals I_1, ..., 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 $i = 1, \ldots, q$. Let p_i be the (unconditional) probability that R selects 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 $\langle I_1, I_2, \ldots, I_m, J_m \rangle$, where $J_m = [1+x(1-m/q), 2+x(1-m/q))$. This is illustrated in Fig. 2.

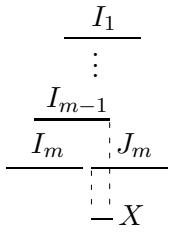

Figure 2: (q, x)-stacking construction.

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

1. *All intervals in $\mathcal{I} \setminus \{I_m\}$ 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 Selection

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 conclude 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 the algorithm RoG (Random_or_Greedy), which handles an arriving interval as follows. If the interval does not overlap any previously presented interval, select it with probability 2/3, else select 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. Therefore, the instance consists 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. Let n_i denote the number of intervals of type i, then 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. 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 solution is the independence number of the path on n vertices, 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 (we omit the proof).

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 for deterministic algorithm 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 $r_1, r_2, \ldots, r_{n-1}$ be a sequence of uniformly random bits. Let the sequence $x_1, x_2, \ldots, x_n$ 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 $I_1, \ldots, I_n$, where $I_i = [x_i, x_i + 2^{n-i})$, for $i = 1, \ldots n$.

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 R and the sequence of intervals chosen by R. 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 of 2. Even accounting for the last interval, which is known to be good, the expected number of accepted intervals $\mathbb{E}[(\sigma)]$ 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 left endpoints. Thus, the bound holds also for the scheduling-by-time model. The adversary in Theorem 6 has also the property of being *transparent* [9] 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 Selection

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. See the top half of Fig. 3. Recall that the expected gain of R on interval I_m is $\mathbb{E}_{\mathsf{R}}[I_m] \leq 1/q'$. Let p be the probability that R selects some interval in $\mathcal{I}' = \mathcal{I} \setminus \{I_m\} = \{I_1, \ldots, I_{m-1}, I_{m+1}\}$. If $p < 2/3$, then we stop the construction. The expected solution size found by R is then $\mathbb{E}_{\mathsf{R}}[\mathcal{I}] \leq p + 1/q'$, 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 $\mathcal{I}'$, 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 $\mathcal{I}_1$ of 2-intervals for R shifted by f_1, where $x = |X_1| = 1/q'$. The first segments are positioned to overlap X_1, where $x = |X_1| = 1/q'$; the second segments are immaterial as long as they do not intersect any previous intervals. This is shown in the bottom left of Fig. 3. We then do an identical construction $\mathcal{I}_2$ 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 $\mathcal{J} = \mathcal{I} \cup \mathcal{I}_1 \cup \mathcal{I}_2$.

Observation 4.1 *1. All intervals in $\mathcal{I}_1$ overlap X_1.*

2. All intervals in $\mathcal{I}_2$ overlap X_2.

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

4. $\mathbb{E}_{\mathsf{R}}[\mathcal{I}_1] \leq (1 - p)(1 + 1/q)$, by Lemma 1 (3) and part 1. of this observation.

It follows that

$$\mathbb{E}_{\mathsf{R}}[\mathcal{J}] = \mathbb{E}_{\mathsf{R}}[I_m] + \mathbb{E}_{\mathsf{R}}[\mathcal{I}'] + \mathbb{E}_{\mathsf{R}}[\mathcal{I}_1] + \mathbb{E}_{\mathsf{R}}[\mathcal{I}_2] \leq 1/q' + p + 2(1 - p)(1 + 1/q) = 2 - p + (4/3 - 2p)q \ .$$

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

4.2 Segments of Two Lengths

In this section, we give a 16-competitive algorithm for 2-Isp where the 2-interval segments have lengths either 1 or d. A lower bound of 6 for $d \gg 1$ is omitted in this version.

Consider the following algorithm A_v, which either schedules (i.e., selects) a given 2-interval, rejects it, or schedules it *virtually*.[1] A virtually scheduled interval does not occupy the resource but blocks other 2-intervals from being scheduled. The length of each segment is either *short* (1) or *long* (d). A 2-interval is *short-short* (*long-long*) if both segments are short (long), respectively, and *short-long* if one is short and the other long. In processing a 2-interval I, A_v applies the following rules, which depend on the availability of the resource.

[1]The term was used before, e.g., in [13].

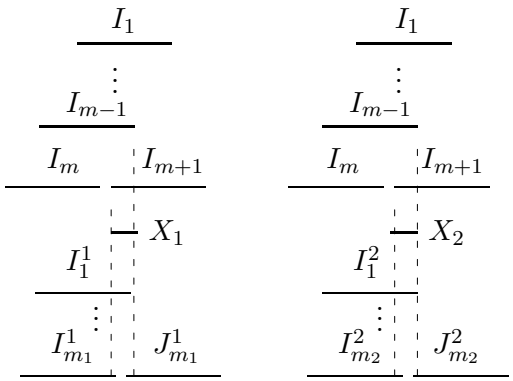

Figure 3: Construction of a lower bound of 3 for unit 2-Isp

1. I **is short-short**. Schedule I greedily (with probability 1).
2. **A long segment of** I **intersects a virtually selected 2-interval**. Do nothing.
3. **Otherwise,** schedule I with probability 1/2 and schedule it virtually with probability 1/2.

Our analysis of A_v uses the following charging scheme. Let S_{OPT} be an optimal solution and S_{A_v} the set of 2-intervals selected by A_v. For any $I \in S_{A_v}$ and $J \in S_{\mathsf{OPT}}$, we assign $w(I,J) = 1/4 \cdot t(I,J)$, where $t(I,J)$ is the number of endpoints of I that intersect with J. In particular, $w(I,J) = 0$ when I and J do not overlap or if segments of I properly contain segments of J. Also, $w(J,J) = 1$. Since each 2-interval has 4 endpoints, and the 2-intervals in S_{OPT} are disjoint, it follows that $\sum_{J \in S_{\mathsf{OPT}}} w(I,J) \leq 1$. Intuitively, we distribute the value that A_v receives for selecting I among the 2-intervals in S_{OPT} that intersect it. Let $w(\mathit{bucket}(J)) = \sum_{I \in S_{A_v}} w(I,J)$, for $J \in S_{\mathsf{OPT}}$. To show that A_v is c-competitive it suffices to prove that, for any $J \in S_{\mathsf{OPT}}$, $\mathbb{E}[w(\mathit{bucket}(J))] \geq 1/c$.

Theorem 8 *Algorithm* A_v *is* 16*-competitive for online* 2-Isp *with segments of length* 1 *and d.*

Proof. Consider an interval $J \in S_{\mathsf{OPT}}$. We shall show that $\mathbb{E}[w(\mathit{bucket}(J)] \geq 1/16$, which yields the theorem. The argument is based on considering the various possible configurations of intervals overlapping J that were presented before J. In what follows, we shall say that an interval was *addressed* if it precedes J, overlaps J, and was either scheduled or virtually scheduled, i.e. was not blocked when presented. We say that I *dominates* J if a short segment of J is properly contained in a long segment of I.

We consider cases depending on the lengths of J's segments.

J **is long-long.** Consider the first interval addressed, I (which is possibly J itself). Then $w(I,J) \geq 1/4$ and I is scheduled with probability at least 1/2. Hence, $\mathbb{E}[w(\mathit{bucket}(J))] \geq 1/2 \cdot 1/4 = 1/8$.

J **is short-long.** Then, there is at most one interval in S_{A_v} that dominates J. With probability at least 1/2, this interval is not selected (so, either virtually selected or blocked). Some other interval I overlapping J (possibly J itself) is then selected with probability at least 1/2, assigning a weight $w(I,J) \geq 1/4$. Thus, $\mathbb{E}[w(\mathit{bucket}(J))] \geq 1/2 \cdot 1/2 \cdot 1/4 = 1/16$.

J **is short-short.** At most two intervals are addressed that dominate J (if they dominate the same segment of J, then the latter interval is blocked by the former). With probability at

least 1/4, neither of them are selected. With probability 1, some other interval I intersecting J (possibly J itself) is selected, since J is short-short. A weight of at least $w(I,J) \geq 1/4$ is transferred. Hence, $\mathbb{E}[w(bucket(J))] \geq 1/4 \cdot 1 \cdot 1/4 = 1/16$.

■

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 $[2^{i-1}, 2^i)$, $1 \leq i \leq K$. Partition the second segments similarly into K groups. 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 length of the short segment is in $[1,2)$ and the long segment in $[d, 2d)$. A_v makes scheduling decisions as before, using the new definitions of 'short' and 'long' segments.

Theorem 9 *Algorithm A_v is 24-competitive for 2-Isp instances with segments of two types: short with lengths in $[1,2)$, and long with lengths in $[d, 2d)$.*

Proof. Each interval I intersects now at most 6 intervals in S_{OPT} that it does not dominate. For instance, a long segment can now contain one long segment from S_{OPT} and properly overlap two other segments. Thus, we change the charging scheme to $w(I,J) = 1/6 \cdot t(I,J)$. The rest of the proof of Theorem 8 is unchanged. ■

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 group. 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 *a priori unknown*, consider algorithm $\tilde{\mathsf{A}}_{vg}$, which proceeds as follows.[2] 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. Thus, each group has an index $i \in \{1, \ldots, \lceil \log \Delta \rceil^2\}$. The algorithm chooses randomly at most one group and selects only 2-intervals from that group, using algorithm A_v. Define

$$c_i = \frac{1}{\zeta(1+\epsilon/2) i^{1+\epsilon/2}}, \quad \text{and} \quad p_i = \frac{c_i}{\Pi_{j=1}^{i-1}(1-p_j)}, \tag{2}$$

where $\zeta(x) = \sum_{i=1}^{\infty} i^{-x}$ is the Riemann zeta function. Recall that $\zeta(x) < \infty$, for $x > 1$.

If a given 2-interval belongs to a new group i, and none of the groups $1, 2, \ldots, i-1$ has been selected, then group i is chosen with probability p_i and rejected with probability $1 - p_i$. If a given 2-interval belongs to an already selected group i, it is scheduled using algorithm A_v; if the given

[2] W.l.o.g., we assume that Δ is an integral power of 2.

2-interval belongs to an already rejected group then it is rejected. Note that by the definition of p_i, as given in (2), it follows that c_i is the unconditional probability that $\tilde{\mathsf{A}}_{vg}$ chooses the i-th group.

In analyzing $\tilde{\mathsf{A}}_{vg}$ we first show that the values p_i form valid probabilities, and that the c_i values give a probability distribution.

Lemma 11 $\sum_{i=1}^{\infty} c_i = 1$. *Also,* $p_i \leq 1$, *for all* $i \geq 1$.

Proof. Observe that $\sum_{i=1}^{\infty} c_i = \frac{1}{\zeta(1+\epsilon/2)} \sum_{i=1}^{\infty} \frac{1}{i^{1+\epsilon/2}} = 1$, proving the first half of the lemma. It follows that $c_i \leq 1 - \sum_{j=1}^{i-1} c_j$. To prove the second half of the lemma, it suffices then to prove the following claim that

$$p_i = \frac{c_i}{1 - \sum_{j=1}^{i-1} c_j}, \tag{3}$$

for each $i \geq 1$. We prove the claim by induction on i. The base case then holds since $p_1 = c_1$. Suppose now that

$$p_{k-1} = \frac{c_{k-1}}{1 - c_1 - c_2 - ... - c_{k-2}} \tag{4}$$

then using (2) we have that

$$p_k = \frac{c_k}{c_{k-1}} \cdot \frac{p_{k-1}}{1 - p_{k-1}} \ .$$

Plugging in the value of p_{k-1} in (4) we get the claim. ■

Theorem 12 $\tilde{A}_{vg}$ *is* $O(\log^{2+\epsilon} \Delta)$*-competitive for* 2-ISP *with intervals of various lengths, where* Δ *is unknown in advance.*

Proof. Let S_i denote the set of 2-intervals in group i, $1 \leq i \leq \log^2 \Delta$.

The probability that $\tilde{\mathsf{A}}_{vg}$ chooses any given group S_i is at least $c_{\log^2 \Delta}$. After selecting the group, $\tilde{\mathsf{A}}_{vg}$ uses A_v to schedule the 2-intervals in the group. For a given group, S_i, we have:

$$\mathbb{E}[\tilde{A}_{vg}(S_i)] \geq c_{\log^2 \Delta} \cdot \mathbb{E}[\mathsf{A}_v(S_i)] \geq \frac{1}{\zeta(1+\epsilon/2)(\log \Delta)^{2+\epsilon}} \cdot \frac{1}{12} \cdot \mathbb{E}[OPT(S_i)].$$

Thus, by linearity of expectation $\tilde{\mathsf{A}}_{vg}$ is $O(\log^{2+\epsilon} \Delta)$-competitive. ■

5 Online t-Interval Selection

We show here that any online algorithm for t-ISP has competitive ratio $\Omega(t)$. 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 13 *Any randomized online algorithm for* t-ISP *with unit segments has competitive ratio* $\Omega(t)$.

Proof. Let n be a positive integer. 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 [8] (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 $\langle v_1, v_2, \ldots, v_n \rangle$. Given vertex v_k and the induced subgraph $G[\langle v_1, v_2, \ldots, v_i \rangle]$, form the n-interval I_i by

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

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*, pages 312–320, 1994.

[2] U. T. Bachmann. *Online t-Interval Scheduling.* MSc thesis, School of CS, Reykjavik Univ., Dec. 2009.

[3] U. T. Bachmann, M. M. Halldórsson, and H. Shachnai. Online scheduling intervals and t-intervals. *Full version*, `http://www.cs.technion.ac.il/~hadas/PUB/onint_full.pdf`, 2010.

[4] A. Bar-Noy, R. Canetti, S. Kutten, Y. Mansour, and B. Schieber. Bandwidth allocation with preemption. In *STOC*, pages 616–625, 1995.

[5] R. Bar-Yehuda, M. M. Halldórsson, J. S. Naor, H. Shachnai, and I. Shapira. Scheduling split intervals. In *SODA*, pages 732–741, 2002.

[6] R. Bar-Yehuda and D. Rawitz. Using fractional primal-dual to schedule split intervals with demands. *Discrete Optimization*, 3(4):275 – 287, 2006.

[7] A. Butman, D. Hermelin, M. Lewenstein, and D. Rawitz. Optimization problems in multiple-interval graphs. In *SODA*, 2007.

[8] M. M. Halldórsson, K. Iwama, S. Miyazaki, and S. Taketomi. Online independent sets. *Theoretical Computer Science*, 289(2):953 – 962, 2002.

[9] M. M. Halldórsson and M. Szegedy. Lower bounds for on-line graph coloring. *Theoretical Comput. Sci.*, 130:163–174, Aug. 1994.

[10] H. A. Kierstead and W. T. Trotter. An extremal problem in recursive combinatorics. In *Congr. Numer. 33*, pages 143–153, 1981.

[11] J. Kleinberg and E. Tardos. *Algorithm Design.* Addison Wesley, 2005.

[12] A. W. Kolen, J. K. Lenstra, C. H. Papadimitriou, and F. C. Spieksma. Interval scheduling: A survey. *Naval Research Logistics*, 54:530–543, 2007.

[13] R. J. Lipton and A. Tomkins. Online interval scheduling. In *SODA*, pages 302–311, 1994.

[14] F. Spieksma. On the approximability of an interval scheduling problem. *J. Sched.*, 2:215–227, 1999.

[15] S. Vialette. On the computational complexity of 2-interval pattern matching problems. *Theor. Comput. Sci.*, 312(2-3):223–249, 2004.

[16] G. J. Woeginger. On-line scheduling of jobs with fixed start and end times. *Theor. Comput. Sci.*, 130(1):5–16, 1994.