Interval Selection with Binary Predictions

Christodoulos Karavasilis
University of Toronto
Toronto, Canada
ckar@cs.toronto.edu

ABSTRACT

Following a line of work that takes advantage of vast machine-learned data to enhance online algorithms with (possibly erroneous) information about future inputs, we consider predictions in the context of deterministic algorithms for the problem of selecting a maximum weight independent set of intervals arriving on the real line. We look at two weight functions, unit (constant) weights, and weights proportional to the interval's length. In the classical online model of irrevocable decisions, no algorithm can achieve constant competitiveness (Bachmann et al. [9] for unit, Lipton and Tomkins [26] for proportional). In this setting, we show that a simple algorithm that is faithful to the predictions is optimal, and achieves an objective value of at least $OPT - \eta$, with η being the total error in the predictions, both for unit, and proportional weights.

When revocable acceptances (a form of *preemption*) are allowed, the optimal deterministic algorithm for unit weights is $2k$-competitive [12], where k is the number of different interval lengths. We give an algorithm with performance $OPT - \eta$ (and therefore 1-consistent), that is also $(2k+1)$-robust. For proportional weights, Garay et al. [19] give an optimal $(2\phi+1)$-competitive algorithm, where ϕ is the golden ratio. We present an algorithm with parameter $\lambda > 1$ that is $\frac{3\lambda}{\lambda-1}$-consistent, and $\frac{4\lambda^2+2\lambda}{\lambda-1}$-robust. Although these bounds are not tight, we show that for $\lambda > 3.42$ we achieve consistency better than the optimal online guarantee in [19], while maintaining bounded robustness.

We conclude with some experimental results on real-world data that complement our theoretical findings, and show the benefit of prediction algorithms for online interval selection, even in the presence of high error.

KEYWORDS

online algorithms, predictions, interval selection, scheduling

1 INTRODUCTION

We consider the problem of online *interval selection*, or *interval scheduling* on a single machine, where real-length intervals arrive online, and we must output a set of non-conflicting intervals. Each interval is associated with a weight, and the goal is to maximize the sum of weights of the intervals in the solution. This problem is equivalent to finding a maximum weight independent set in interval graphs. We focus on two weight functions, *unit* (or constant) weights, and *proportional* weights, where the weight of an interval is equal to its length. While interval scheduling is often studied under the real-time assumption where intervals arrive in order of non-decreasing starting times, we consider the generalized version of *any-order* arrivals [12]. In the traditional online model of irrevocable decisions, no algorithm (even randomized) can achieve a constant competitive ratio (Bachmann et al. [9] for unit weights, Lipton and Tomkins [26] for proportional). Because of this, and because some applications permit it, a relaxation of the problem that allows for revocable acceptances has been considered. In this model, every new interval can be accepted by displacing any conflicting intervals in the solution, but every rejection is final. In the area of scheduling, this is sometimes also called *preemption*, although no "restarts" are allowed in our problem. In the offline setting, an optimal solution can be easily found in polynomial time, both for unit, and for proportional weights [23]. The applications of interval scheduling include routing [29], computer wiring [21], project selections during space missions [22], and satellite photography [18]. A more detailed discussion on the applications of interval scheduling can be found in the surveys by Kolen et al. [24] and Kovalyov et al. [25].

Motivated by advancements in machine learning and access to a plethora of data, there has been an effort to equip online algorithms with possibly erroneous predictions about the input instance. Such algorithms are able to achieve much better performance when these predictions are accurate, overcoming some pessimistic bounds of competitive analysis, and helping to bridge the gap between theory and practice. Various classical online problems such as ski rental and non-clairvoyant job scheduling [30], caching [27], facility location [2], metrical task systems [7], and matching [8] have been considered in this model. See Mitzenmacher and Vassilvitskii [28] for a more detailed survey on the topic, and [1] for an online repository of relevant papers. Predictions are also a form of *untrusted advice* (Angelopoulos et al. [3]), a natural extension of the model of online algorithms with advice (Boyar et al. [14]) when the advice is imperfect. Advice research tends to be more information theoretic, focusing on tradeoffs between the number of advice bits and the quality of the solution. Although predictions are often available as offline information, given to the algorithm in advance, we consider a model where a prediction is associated with each input item, and is also given online. This is quite natural and has been considered before for problems such as paging, graph coloring, and packing ([5, 6, 20, 27, 31]). This setting also allows for an oracle to adapt as more of the input is revealed, enabling research where there are different bounds on the quality of later predictions, and allowing one to tailor the predictor algorithm directly [15]. Furthermore, we use binary predictions, which has our model falling in line with work considering limited size, or *succinct* predictions [4, 5, 10].

Related work. Table 1 shows the most relevant existing work in the conventional online setting. In the case of irrevocable decisions, no algorithm (even randomized) can achieve a constant competitive ratio. For the relaxed model of revocable acceptances, we use an asterisk to indicate that the competitive ratio is optimal. In the context of randomized algorithms and revocable acceptances,

Emek at al. [16] give a 6-competitive algorithm for unit weights, while we know of no work improving upon the $(2\phi+1)$-competitive algorithm.

Table 1: Online results without predictions: n is the size of the input, k is the number of different lengths, Δ is the ratio of the longest to shortest interval.

	Unit	*Proportional*
Irrevocable (randomized)	$\Omega(n)$ [9]	$\Omega(\log \Delta)$ [26]
Revocable (deterministic)	$2k^*$ [12]	$(2\phi+1)^*$ [19, 32]

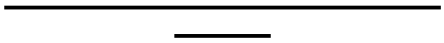
Boyar et al. [13] is the most closely related work to our problem with predictions, and motivated our study. They consider the case of unit weighted intervals on a line graph, and give an optimal deterministic algorithm in the setting of irrevocable decisions with performance $OPT - \eta$ for a different set of predictions and error measure. We extend their work using (possibly adaptive) predictions of limited size, considering an additional weight function of interest, and initiating the study of these problems with revocable decisions.

Structure of the paper. In section 2 we formally define the model, including our predictions and error measure. Section 3 is about the model of irrevocable decisions, whereas in section 4 we allow for revocable acceptances. We conclude with some experiments on real-world data (section 5) that showcase the usefulness of our predictions, and complement our theoretical results.

2 PROBLEM SETTING, DEFINITIONS AND DISCUSSION

In the problem of *interval selection*, an instance consists of a set of intervals arriving on the real line. Each interval is specified by a starting point s_i and an end point f_i, with $s_i < f_i$, and it occupies space $[s_i, f_i)$. The conventional notions of *intersection*, *disjointness*, and *containment* apply. Two intervals can conflict because of a *partial conflict*, or because of *proper inclusion* (figure 1). In the latter case, we say that the smaller (larger) interval is subsumed (subsumes) by the other. Each interval I is also associated with a weight $w(I)$. The goal is to output a set of disjoint intervals of maximum weight. We focus on two types of weight functions, *unit* weights where $w(I) = 1$ for all I, and *proportional* weights where $w(I) = f_i - s_i$. With unit weights, we want to accept as many intervals as possible, whereas with proportional weights, we want to cover as much of the line as possible.

(a) Partial Conflict.



(b) Proper inclusion conflict.

Figure 1: Types of conflicts.

The sequence of the intervals arriving online is chosen by an adversary with full knowledge of the algorithm. In the conventional model of *irrevocable decisions*, each accepted interval is final and will be part of the final solution. A well studied relaxation of the model is allowing for *revocable acceptances*, where any new interval may be accepted by displacing any conflicting intervals in the solution, but every rejection is final. This is sometimes also referred to as *preemption*.

We measure the performance of an online algorithm using *worst case competitive analysis* [11]. Let ALG denote the total weight of the algorithm's solution, and OPT be the total weight of an optimal solution. An algorithm is strictly c-competitive, if for all instances and input permutations, we have that $\frac{OPT}{ALG} \geq c$. We also refer to ALG as the *performance* of the algorithm. Lastly, we will use ALG (respectively OPT) to denote the set of intervals in the algorithm's (optimal) solution. The meaning should always be clear from context.

Predictions. Every interval is associated with a binary prediction that becomes known at the time of the interval's arrival, denoting whether or not that interval is part of some fixed optimal solution. More precisely, $Prd(I) = 1$ means interval I is predicted to be optimal, and $Prd(I) = 0$ means that it is predicted to not be optimal. Let $\mathcal{I}$ denote the set of all intervals in the instance. We define $\overrightarrow{\mathbf{p}} = (Prd(I_1), Prd(I_2), ..., Prd(I_{|\mathcal{I}|}))$ to be the binary vector of all the online predictions. Let also $\overrightarrow{\mathbf{p}_+}$ be the predictions vector when all predictions are accurate, and $\overrightarrow{\mathbf{p}_-} = \overline{\overrightarrow{\mathbf{p}_+}}$.

Error. Every inaccurate prediction introduces an amount of error. Let $\eta(I)$ be the amount of error introduced by interval I. If the prediction of I was accurate, we define $\eta(I) = 0$. If I was wrongly predicted to be non-optimal, let $\eta(I) = w(I)$, and if I was wrongly predicted to be optimal, and C is the set of optimal intervals I conflicts with, let $\eta(I) = \sum_{J \in C} w(J) - w(I)$. Let the total error be $\eta = \sum_{I \in \mathcal{I}} \eta(I)$. One may fix any optimal solution to measure the error against. We use η_{max} to denote the maximum possible error, i.e. $\eta_{max} = \eta$ when $\overrightarrow{\mathbf{p}} = \overrightarrow{\mathbf{p}_-}$.

A common approach to evaluate algorithms that use predictions is to focus on an algorithm's *consistency* and *robustness*. We say that an algorithm is γ-consistent if it is γ-competitive, whenever the predictions are accurate, i.e. $\overrightarrow{\mathbf{p}} = \overrightarrow{\mathbf{p}_+}$. We say that an algorithm is ζ-robust, if it is ζ-competitive regardless of the accuracy of the

predictions. There is usually a trade-off between consistency and robustness, and the goal is to design algorithms with consistency close to 1, and robustness that is not far worse than the competitiveness of the best predictionless, online algorithm.

Most of our proofs work by using a *charging argument*, where we map the weight of optimal intervals to the weight of intervals taken by the algorithm, and sometimes error. This charging is usually defined in an online manner, and throughout the execution of the algorithm, we use $\Phi(I)$ to refer to the total amount of charge to interval I. In the model of revocable acceptances, we will distinguish between *direct*, and *transfer* charging. Transfer charging (TC) occurs at the moment a new interval is accepted by replacing existing intervals, and refers to the amount of charge it inherits because of this. Direct charge (DC) takes place afterwards, whenever an interval causes optimal intervals to be rejected. We use $TC(I)$ (respectively $DC(I)$) to denote the amount of transfer (direct) charge of an interval, with $\Phi(I) = TC(I) + DC(I)$.
We also use the notion of a *predecessor trace*, which is analogous to Woeginger's [33] *predecessor chain* in th real-time model. If I is an interval in the algorithm's solution, the **predecessor trace** $\mathcal{P}$ of I is the maximal list of intervals $(P_1, P_2, .., P_k = I)$, such that P_i was at some point accepted by the algorithm, but was later replaced by P_{i+1}.

3 IRREVOCABLE ACCEPTANCES

In utilizing access to predictions, a natural algorithm to first consider is one that simply *follows the predictions*. We therefore present algorithm 1, which is the main subject of this section.

Algorithm 1 `Naive`

On the arrival of I:
$I_s \leftarrow$ Set of intervals currently in the solution conflicting with I
if $Prd(I) = 1$ and $I_s = \emptyset$ **then**
  Take I

Unit weights. We first consider the case of unit weights, and prove the following (positive) result on the performance of algorithm `Naive`.

Theorem 3.1. *Algorithm Naive achieves $ALG \geq OPT - \eta$ for interval selection with unit weights.*

Proof. The proof works by mapping optimal intervals to error, and to intervals taken by the algorithm, given that every missed optimal interval can be associated with at least one unit of error. Let OPT be an optimal solution, and let ALG be the algorithm's solution. For each unit of error, we define an error element h. Let $H = \{h_1, ..., h_\eta\}$ be the set of error elements. Let $H_I \subseteq H$, be the set of error elements corresponding to the error $\eta(I)$. It holds that $H_I \cap H_J = \emptyset$ for any two distinct intervals I, J, and $\bigcup_I H_I = H$.

We define an injective mapping $F : OPT \rightarrow ALG \cup H$ as follows: Let I_{opt} be an optimal interval. If I_{opt} is taken by the algorithm, it is mapped onto itself. If I_{opt} is not taken by the algorithm, there are two possibilities. The first possibility is that $Prd(I_{opt}) = 1$, but I_{opt} conflicted with another interval I_c taken by the algorithm. For I_c to have been taken, it means that $Prd(I_c) = 1$, and $|H_{I_c} \cup \{I_c\}| > 0$ because I_c conflicts with I_{opt}. In this case we map I_{opt} to an error element $h_c \in H_{I_c}$, or I_c itself. Even if more optimal intervals were not taken because they conflicted with I_c, there will be enough distinct elements in $H_{I_c} \cup \{I_c\}$ to map them to.

The second possibility is that I_{opt} was not taken, because $Prd(I_{opt}) = 0$. In this case, we have that $|H_{I_{opt}}| = 1$, and we can map I_{opt} to the error element of its own prediction. In conclusion, we have that $ALG + |H| \geq OPT$, and we get the desired bound. □

Corollary 3.2. *Algorithm Naive is 1-consistent.*

Theorem 3.3. *For every deterministic algorithm, there exist a unit weights instance and predictions, such that $ALG = OPT - \eta$.*

Proof. Let an interval I_{big} arrive first, with $Prd(I_{big}) = 0$. If the algorithm rejects it, no more intervals arrive, and we have $ALG = 0$, $OPT = 1$, $\eta = 1$. If the algorithm takes I_{big}, two non-conflicting intervals arrive next, I_1 and I_2, both subsumed by I_{big}, with $Prd(I_1) = 0$ and $Prd(I_2) = 1$. In this case, we have $ALG = 1$, $OPT = 2$, $\eta = \eta(I_1) = 1$. In both cases, the equality holds. One can repeat this construction for an asymptotic result. □

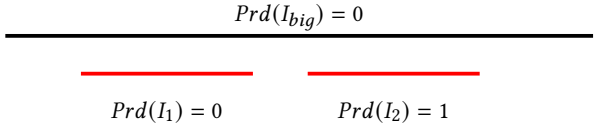


Figure 2: Instance of theorem 3.3.

Corollary 3.4. *Algorithm Naive is optimal for unit weights in the model of irrevocable acceptances.*

Boyar et al. [13] were the first to consider the problem of interval selection with unit weights and irrevocable decisions, and they get the same (syntactically) performance, using a different algorithm, and a different set of predictions and error measure. In comparing our result to theirs, we note that our predictions are information theoretically strictly weaker than theirs[1], and can in fact easily be extracted from theirs, allowing our algorithms to operate in their model. Furthermore, their predictions-following algorithm is enhanced with a *greedy* aspect in order to achieve this optimal performance. As we will see in section 5, experimental results on real-world data suggest that for some error ranges, pure *greediness* is arguably a more important attribute than the use of predictions for getting a good solution, and the combination of both in the context of revocable acceptances works best.

We will now show that with **proportional weights**, algorithm `Naive` achieves the same performance bounds as in the case of unit weights.

Theorem 3.5. *Algorithm Naive achieves $ALG \geq OPT - \eta$ for interval selection with proportional weights.*

[1]Their predictions consist of the entire input instance given in advance.

Proof. Without loss of generality, we assume integral lengths of intervals, and later explain how to generalize to real lengths. We discretize the weight of intervals into weight units, and define a weight element w for each unit of weight. Let $W_I = \{w_1, ..., w_{w(I)}\}$ be the set of weight elements corresponding to the weight of interval I, with $W_I \cap W_J = \emptyset$ for any two distinct intervals I, J . Let $W_{opt} = \bigcup_{I \in OPT} W_I$, and $W_{alg} = \bigcup_{I \in ALG} W_I$. The sets H and H_I are defined as for the unweighted algorithms. Lastly, let $C_{opt}(I)$ be the set of optimal intervals in OPT that conflict with interval I.

We argue for the existence of an injective mapping $F : W_{opt} \rightarrow W_{alg} \cup H$ as follows: Let I_{opt} be an optimal interval. If I_{opt} is taken by the algorithm, we map the elements of $W_{I_{opt}}$ to their corresponding elements in W_{alg}. If I_{opt} is not taken by the algorithm, there are two cases. The first case is that I_{opt} did not conflict with any interval in the solution, but $Prd(I_{opt}) = 0$. In this case, we know that $\eta(I_{opt}) = |H_{I_{opt}}| = w(I_{opt})$, and we can map the weight elements of I_{opt} to error elements in $H_{I_{opt}}$.

The other case is that I_{opt} conflicted with at least one interval in the solution. Let that conflicting interval be I_c. It could also be that $Prd(I_{opt}) = 0$, and $H_{I_{opt}}$ would have error elements we can use, but we will assume the worst case of $Prd(I_{opt}) = 1$. In this case, it could be $|W_{I_{opt}}| > |H_{I_c}|$, and we cannot map the weight elements to error elements exclusively. We can, however, map the optimal weight elements, to elements in $H_{I_c} \cup W_{I_c}$. To see that there will always be sufficiently many unmapped elements, notice that $|H_{I_c} \cup W_{I_c}| \geq |\bigcup_{I \in C_{opt}(I_c)} W_I|$. This is because $|H_{I_c}| = |\bigcup_{I \in C_{opt}(I_c)} W_I| - |W_I|$, and $W_{I_c} \cap H_{I_c} = \emptyset$ always holds. We conclude that $|W_{alg}| + |H| \geq |W_{opt}|$, and we get the desired bound.

To adapt the proof to real lengths, instead of considering sets of error and weight elements, we can define a transport plan using two transport matrices H and W of size $OPT \times ALG$. H_{ij} (respectively W_{ij}) corresponds to the (real) amount of weight, or mass, mapped from $I_i \in OPT$ to the amount of error (resp. weight) introduced by $I_j \in ALG$. We can define these matrices such that for $1 \leq i \leq OPT$, $\sum_{1 \leq j \leq ALG} H_{ij} + W_{ij} = w(I_i)$, for $1 \leq j \leq ALG$, we have that $\sum_{1 \leq i \leq OPT} H_{ij} \leq \eta(I_j)$ and $\sum_{1 \leq i \leq OPT} W_{ij} \leq w(I_j)$. □

Theorem 3.6. *For every deterministic algorithm, there exist a proportional weights instance and predictions, such that $ALG = OPT - \eta$.*

Proof. Let I_1 arrive first with $Prd(I_1) = 0$. If the algorithm doesn't accept I_1, no more intervals arrive, and we have that $ALG = 0$, $OPT = w(I_1)$, and $\eta = w(I_1)$. If the algorithm accepts I_1, let two intervals I_2 and I_3 arrive next, with $w(I_2) = w(I_1)$, $Prd(I_2) = 1$, $w(I_3) = 2w(I_1)$ and $Prd(I_3) = 0$. In this case we have $ALG = w(I_1)$, $OPT = 3w(I_1)$, and $\eta = \eta(I_3) = 2w(I_1)$. In both cases the equality holds. One can repeat this construction for an asymptotic result. □

Corollary 3.7. *Algorithm Naive is optimal for proportional weights in the model of irrevocable acceptances.*

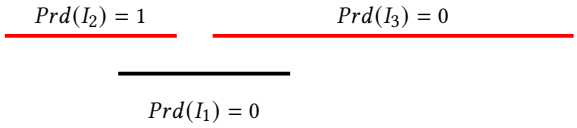


Figure 3: Instance of Theorem 3.6, with $w(I_2) = w(I_1), w(I_3) = 2w(I_1)$.

4 REVOCABLE ACCEPTANCES

Given the difficulty of the problem(s) in the conventional online model, we now consider the case where acceptances are revocable, but rejections are final, a relaxation of the model that is commonly studied for the problem of interval selection. A new interval can now always be accepted by displacing any intervals in the solution conflicting with it. For unit weights, Borodin and Karavasilis [12] give an optimal algorithm that is $2k$-competitive, where k is the number of distinct interval lengths. We will refer to this algorithm of [12] as the *BK2K* algorithm. *BK2K* is a greedy algorithm, always accepting a new interval when there is no conflict, and whenever a conflict exists, the new interval is accepted only if it is properly included in an interval currently in the solution. We use that as the base logic for our predictions algorithm, and add one more replacement rule, which accepts a new interval I that is only involved in partial conflicts, if $Prd(I) = 1$. Furthermore, an interval accepted by that rule gets marked, to make sure it cannot be replaced by that rule again. We call this algorithm `Revoke-Unit` 2. Interestingly, this rule of locally *following the predictions once*, suffices to give us 1-consistency.

Algorithm 2 `Revoke-Unit`

$M \leftarrow \emptyset$ ▷ Set of marked intervals
$S \leftarrow \emptyset$ ▷ Solution set
On the arrival of I:
$I_s \leftarrow$ Set of intervals currently in the solution conflicting with I
if $I_s = \emptyset$ or ($I_s = \{I'\}$ and $I \subset I'$) **then**
  if $I' \in M$ **then**
    $M \leftarrow M \cup \{I\}$
  $S \leftarrow S \cup \{I\} \setminus \{I'\}$ ▷ Take I and discard I' if necessary
else if I is only involved in partial conflicts **and** $Prd(I) = 1$ **and** $I_s \cap M = \emptyset$ **then**
  $S \leftarrow S \cup \{I\} \setminus I_s$ ▷ Take I and discard conflicting intervals
  $M \leftarrow M \cup \{I\}$

Theorem 4.1. *Algorithm 2 achieves $ALG \geq OPT - \eta$.*

Proof. We follow the same approach as in the proof of Theorem 3.1, mapping optimal intervals to intervals taken by the algorithm, and to error. The main difference is that because of revoking, this mapping might be redefined throughout the execution of the algorithm. As before, we let H be the set of error elements, and $H_I \subseteq H$ be the set of error elements introduced by $\eta(I)$. Let I_{opt} be an optimal interval. We will define an injective mapping $F : OPT \rightarrow ALG \cup H$ as follows: If I_{opt} is taken by the algorithm, it is initially mapped onto itself. If I_{opt} is later replaced, it must be because of a partial conflict (w.l.o.g. no interval is subsumed by an

optimal interval) with a new interval I' with $Prd(I') = 1$. In this case, I_{opt} will be mapped to an error element in $H_{I'}$, or if no further optimal intervals that conflict with I are yet to arrive, it will be mapped to I. In both, subcases it will never be remapped.
Consider now the case of I_{opt} being rejected upon arrival. This can only happen if it is involved in (at most two) partial conflicts. There are two possible cases. The first case is that $Prd(I_{opt}) = 0$, and therefore $|H_{I_{opt}}| = 1$, in which case we map I_{opt} to the error element of its own prediction. The second case is that $Prd(I_{opt}) = 1$, but at least one of the conflicting intervals was marked. Let I_c be one of the marked, partially conflicting intervals. If I_c was marked by being taken through a partial-conflict replacement, it means that $Prd(I_c) = 1$, and $|H_{I_c} \cup \{I_c\}| > 0$, in which case we can map I_{opt} to an element $h_c \in H_{I_c} \cup \{I_c\}$.
If I_c was instead marked by a proper-inclusion-replacement, we trace the original interval that got marked through a partial-conflict-replacement. Call that interval $I_c^{'}$. It holds that $I_c^{'}$ conflicts with I_c, and therefore also conflicts with I_{opt}. Moreover, for $I_c^{'}$ to have been accepted, it must be that $Prd(I_c^{'}) = 1$ and $|H_{I_c^{'}} \cup \{I_c^{'}\}| > 0$. In this case, we map I_{opt} to an element $h_{c'} \in H_{I_c^{'}} \cup \{I_c^{'}\}$. In conclusion, we have that $ALG + |H| \geq OPT$, and we get the desired bound. □

We note that the performance of algorithm 2 on the instance of theorem 3.3 is exactly equal to $OPT - \eta$, and we get the following lemma.

Lemma 4.2. *The performance of algorithm 2 cannot be better than $OPT - \eta$.*

We next show that the robustness of algorithm `Revoke-Unit` nearly matches the optimal online guarantee.

Theorem 4.3. *With at most k distinct interval lengths, algorithm 2 is $(2k + 1)$-robust.*

Proof. We use a charging argument and show that an interval taken by the algorithm can be charged by at most $2k + 1$ optimal intervals. As soon as an optimal interval arrives, we map it to an interval already taken by the algorithm or itself. When an interval is replaced during the execution, all optimal intervals charged to it up to that point, will now be charged to the new interval that was accepted. We build upon the proof of Theorem 3.2 in [12]. In the case of the *BK2K* algorithm ([12]), it is true that for every predecessor trace $\mathcal{P}$, and consecutive intervals $(I_i, I_{i+1}) \in \mathcal{P}$, $\Phi(I_{i+1}) \leq \Phi(I_i) + 2$, and the length of every predecessor trace is at most k. While the former is still true for algorithm `Revoke-Unit`, the latter is not, and that is because we have an additional replacement rule. However, we argue that for every $I_i \in \mathcal{P}$, if I_j, $j > i$ is the next interval in the trace that was accepted through proper-inclusion, it is true that $TC(I_j) \leq TC(I_i) + 3$.

Figure 4 shows how to maximize charge on the event of a partial-conflict replacement. Before being replaced by a partially-conflicting interval, I_1 can be directly charged by at most two optimal intervals (I_{opt}^1, I_{opt}^2), one on each side. After I_2 replaces I_1, it can also be directly charged by two optimal intervals, but only if I_{opt}^2 was not charged to I_1 earlier. In other words, if I_2 is directly charged by two new intervals, it means that I_1 was directly charged by at most one, concluding that $\Phi(I_2) \leq TC(I_1) + 3$.

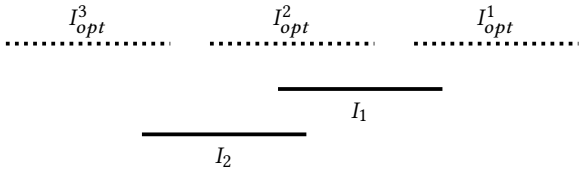


Figure 4: Maximum charge through partial-conflict replacement.

Finally, notice that because the mark of an interval carries over when it is replaced, the event of a partial-conflict replacement can occur at most once in each predecessor trace, and excluding at most one subsequence $(I_i, I_r, I_j) \in \mathcal{P}$ where $TC(I_j) \leq TC(I_i) + 3$, it holds that for $(I_b, I_{b+1}) \in \mathcal{P}$, $\Phi(I_{b+1}) \leq \Phi(I_b) + 2$, giving us a worst case competitive ratio of $2k + 1$. □

Corollary 4.4. *With at most k distinct interval lengths, and predictions with total error η, Algorithm Revoke-Unit achieves $ALG \geq \max\{OPT - \eta, \frac{OPT}{2k+1}\}$.*

Notice how we can choose not to carry over the mark when proper-inclusion replacement occurs, and get a $3k$-robust algorithm. Such an algorithm is prone to follow the prediction more often, and it can outperform `Revoke-Unit` for some small values of error caused by adversarial predictions.

We now look at the case of proportional weights. In the conventional online setting, Garay et al. [19] give a $2\phi + 1 \approx 4.236$−competitive algorithm, while Tomkins [32] gives a matching lower bound. They call their optimal algorithm `LR` (for *length of route*), and we include it here for completeness. Unlike the case of unit weights, we now want to accept intervals that occupy as much of the line as possible. Algorithm `LR` works greedily by always accepting a new interval with no conflicts, and when there are conflicts, it accepts the new interval if its length is at least ϕ times greater than the largest conflicting interval. More generally, using parameter $\beta \geq \phi$, we have the following lemma:

Lemma 4.5 (Garay et al. [19]). *Algorithm `LR` with parameter $\beta \geq \phi$ is $(2\beta + 1)$-competitive for the problem of interval selection with proportional weights.*

Algorithm 3 LR [19]

Parameter $\beta = \phi$ ▷ optimal value for parameter β
On the arrival of I:
$I_s \leftarrow$ Set of intervals currently in the solution conflicting with I
if $w(I) > \beta \cdot \max\{w(J) : J \in I_s\}$ **then**
  Accept I and displace conflicts
  Return

Instead of using algorithm `LR` as the base of our predictions algorithm, we consider a slightly modified version, which we refer to

as LR′, and which compares the weight of the new interval to the sum of the weights of the conflicting intervals, instead of looking only at the longest interval. Although we do not know the exact performance of algorithm LR′ in the online model, we conjecture it is also $(2\phi + 1)$-competitive.

In trying to utilize predictions in the case of proportional weights, we first make the following observations:

Observation 4.6. *1-consistency is unattainable while maintaining bounded robustness.*

Proof. To be 1-consistent, the algorithm must be able to replace an interval with an arbitrarily smaller one that is part of the optimal solution. The adversary could then stop the instance, forcing arbitrarily bad robustness. □

Observation 4.7. *In order to have bounded robustness, it must be that a new interval that is sufficiently large (small[2]) must always be accepted (rejected).*

Definition 4.8 (α−increasing). An α-increasing algorithm never accepts a new conflicting interval that is less than α times the longest interval it conflicts with.

Lemma 4.9. *An α-increasing algorithm (greedy or non-greedy), cannot be better than $(2\alpha + 1)$-consistent.*

Proof. Let an interval I_1 arrive first. Let I_2 and I_3 be intervals that partially conflict with I_1 on either side, and $w(I_2) = w(I_3) = \alpha \cdot w(I_1) - \epsilon$. Let I_4 with $w(I_4) = w(I_1) - 2\epsilon$ be an interval that is fully subsumed by I_4. This instance is depicted in figure 5. The algorithm will never replace I_1, while the optimal solution is made of $\{I_2, I_3, I_4\}$. □

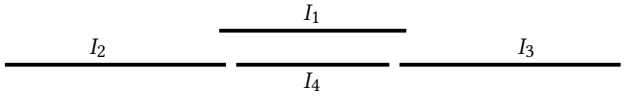


Figure 5: Consistency bound for α-increasing algorithms.

Algorithm LR is a ϕ-increasing algorithm, while the algorithm by Woeginger [33] for the real-time model is 2-increasing. Our predictions algorithm 4 Revoke-Proportional is 1-increasing. The algorithm works like LR′, with one additional replacement rule that accepts a new interval that is predicted to be optimal, even if it is not sufficiently larger than what it conflicts with. More precisely, if a new interval is predicted to be optimal and is at least as big as the sum of the weights of the intervals it conflicts with, and none of the conflicting intervals were predicted to be optimal, it will be accepted through the predictions rule. The algorithm takes a parameter $\lambda > 1$, which can be thought of as an indicator of how much the predictions are trusted. As λ increases, the consistency bound improves.

Theorem 4.10. *Algorithm Revoke-Proportional is $\frac{3\lambda}{\lambda-1}$-consistent.*

[2]More accurately, an interval reducing ALG sufficiently much.

Algorithm 4 Revoke-Proportional Parameter: $\lambda > 1$

```
On the arrival of I:
I_s ← Set of intervals currently in the solution conflicting with I
Let w_c = Σ_{J∈I_s} w(J)        ▷ Total weight of conflicting intervals
if w(I) ≥ λ · w_c then                          ▷ Main replacement rule
    Accept I and displace conflicts
    Return
else if Prd(I) = 1 then                          ▷ Predictions rule
    if (w(I) ≥ w_c and |{J : J ∈ I_s and Prd(J) = 1}| = ∅) then
        Accept I and displace conflicts
        Return
```

Proof. We consider the optimal solution OPT consistent with the fully accurate predictions. We will show that throughout the execution of the algorithm, we have that $\Phi(I) \leq \mu \cdot w(I)$, for every I in the current solution. In the end, we have that $\sum_{I \in ALG} \Phi(I) = OPT$, giving us the μ-consistency of the algorithm.

As in the proof of theorem 4.3, we consider the notions of *transfer charge* (TC), and *direct charge* (DC). We can express $\Phi(I) = TC(I) + DC(I)$. A transfer charge occurs whenever accepting a new interval I replaces intervals currently in the solution. In that case, the total charge of those conflicting intervals is passed on as transfer charge to I. Any additional charge to I after its acceptance is through direct charge, namely rejection of subsequent optimal intervals conflicting with I. We will write $DC_J(I)$ to denote the amount of direct charge from interval J to interval I. Whenever an optimal interval is accepted, we consider its weight being directly charged to itself, and it cannot be directly charged again.

Whenever an optimal interval is rejected upon arrival, we charge its weight to the intervals it conflicts with, with its weight being distributed to all its conflicting intervals, in proportion to their weight. Specifically, let I_o be the newly arrived optimal interval that is rejected, and I_s denote the set of conflicting intervals. Each interval $J \in I_s$ is directly charged $DC_{I_o}(J) = w(I_o)\frac{w(J)}{w_c} \leq w(I_o)$. Furthermore, for an optimal interval to have been rejected, it must be that even the predictions rule failed, and because the predictions are accurate, it must have failed because $w(I_o) < w_c$. Because of this, we get that $w(I_o)\frac{w(J)}{w_c} \leq w(J)$, and therefore $DC_{I_o}(J) \leq \min\{w(I_o), w(J)\}$. An interval $I \in ALG$ can be directly charged by at most three different types of optimal intervals: 1) smaller intervals that are subsumed by it, 2) an optimal interval partially conflicting on the left, and 3) an optimal interval partially conflicting on the right. In the case of smaller optimal intervals subsumed by I, the total amount of direct charge from those intervals can be at most $w(I)$. Given that each of the two possible partially conflicting intervals can directly charge I at most $w(I)$, we conclude that for every $I \in ALG$:

$$DC(I) \leq 3w(I) \tag{1}$$

We omitted the case where the rejected optimal interval subsumes I, because in that case $DC(I) = w(I)$ and 1 holds trivially. We now focus on the total amount of charge on any interval $I \in ALG$. Let:

$$\mu = \frac{3\lambda}{\lambda - 1}$$

We want to make sure that throughout the execution of the algorithm, $\Phi(I) \leq \mu \cdot w(I)$. Before any interval is accepted through replacement, intervals in the solution could have only been directly charged through rejected optimal intervals, and because of 1, and the fact that $\lambda > 1$, our desired bound holds. We now consider all the cases of an interval being accepted through replacement.

Case 1: I is an optimal interval and it is accepted through the predictions rule. In this case we have that $DC(I) = w(I)$, and we need to look at $TC(I)$. Let L_c and R_c denote the intervals (if any) that I is partially conflicting with on the left and on the right respectively, and let M_c denote the set of intervals that I subsumes. We know that all of these conflicting intervals are not optimal, and they were accepted through the algorithm's main rule. First, notice that for all $J \in M_c$, $DC(J) = 0$, and $\Phi(J) = TC(J) \leq \frac{\mu}{\lambda} \cdot w(J)$. Moreover, L_c and R_c had not yet been directly charged by a partially conflicting optimal interval on one side, and therefore we have that $\Phi(L_c) \leq \frac{\mu}{\lambda} \cdot w(L_c) + 2w(L_c)$, and similarly $\Phi(R_c) \leq \frac{\mu}{\lambda} \cdot w(R_c) + 2w(R_c)$. Putting everything together:

$$\begin{aligned} TC(I) &= \sum_{J \in M_c} \Phi(J) + \Phi(L_c) + \Phi(R_c) \\ &\leq \frac{\mu}{\lambda} \cdot w_c + 2(w(L_c) + w(R_c)) \\ &\leq \left(\frac{\mu}{\lambda} + 2\right) w_c \\ &\leq \left(\frac{\mu}{\lambda} + 2\right) w(I) \end{aligned}$$

The last inequality being true from the fact that the main predictions rule is satisfied. Given also that $DC(I) = w(I)$, we get that $\Phi(I) \leq (\frac{\mu}{\lambda} + 2)w(I) + w(I) = (\frac{\mu}{\lambda} + 3)w(I)$. With our choice of μ, we have:

$$\begin{aligned} \Phi(I) &\leq \left(\frac{\frac{3\lambda}{\lambda-1}}{\lambda} + 3\right) w(I) \\ &= \left(\frac{3\lambda}{\lambda - 1}\right) w(I) \end{aligned}$$

Case 2: I is an optimal interval and it is accepted through the algorithm's main rule. This is similar to case 1, with $DC(I) = w(I)$ and $w(I) \geq \lambda \cdot w_c$. The same analysis gives us $TC(I) \leq \left(\frac{\mu}{\lambda} + 2\right) \frac{w(I)}{\lambda}$, and because $\lambda > 1$, the same bound holds.

Case 3: I is not an optimal interval and it is accepted through the algorithm's main rule. In this case we have that $DC(I) \leq 3w(I)$, and we get that

$$\begin{aligned} \Phi(I) &\leq \sum_{J \in I_s} \Phi(J) + 3w(I) \\ &\leq \frac{\mu}{\lambda} \cdot w(I) + 3w(I) \\ &= \left(\frac{3\lambda}{\lambda - 1}\right) w(I) \end{aligned}$$

In conclusion, we have that throughout the execution of the algorithm, for $I \in ALG$, $\Phi(I) \leq \frac{3\lambda}{\lambda-1} w(I)$, and therefore $\frac{OPT}{ALG} \leq \frac{3\lambda}{\lambda-1}$. □

We see that as $\lambda \to \infty$, the algorithm's consistency goes to 3. We now look at the robustness of algorithm `Revoke-Proportional`.

Theorem 4.11. *Algorithm Revoke-Proportional is* $\frac{4\lambda^2+2\lambda}{\lambda-1}$*-robust.*

Proof. The argument is similar to the proof of theorem 4.10. Both *direct*, and *transfer* charging work the same way as before. Let $\mu = \frac{2\lambda^2+3\lambda+1}{\lambda-1}$, and $\delta = 2\lambda + 1$. We will show that that for every $I \in ALG$, $\Phi(I) \leq (\mu + \delta) \cdot w(I) = \frac{4\lambda^2+2\lambda}{\lambda-1} w(I)$.

Notice first that the upper bound on direct charging is not as good as before. More precisely, with I_o being a newly arrived optimal interval that will be rejected and I_s being its conflicting intervals currently in the solution, we have that for every $J \in I_s$, $DC_{I_o}(J) = w(I_o)\frac{w(J)}{w_c} \leq \lambda \cdot w(J)$. More generally, $DC_{I_o}(J) \leq \min\{w(I_o), \lambda \cdot w(J)\}$. As before, given the three different possible types of conflicts, we have that:

$$DC(I) \leq (2\lambda + 1)w(I) \tag{2}$$

We can now bound the total amount of charge on every interval in the algorithm's solution, throughout its execution. Before any replacement happens, the bound $\Phi(I) \leq (\mu+\delta) \cdot w(I)$ holds trivially.

Case 1: Interval I is accepted through the algorithm's main rule. We get that:

$$\begin{aligned} \Phi(I) &\leq (\mu + \delta) \cdot w_c + (2\lambda + 1) \cdot w(I) \\ &\leq (\mu + \delta) \cdot \frac{w(I)}{\lambda} + (2\lambda + 1) \cdot w(I) \\ &= \left(\frac{\mu + \delta}{\lambda} + 2\lambda + 1\right) w(I) \\ &= \left(\frac{\frac{4\lambda^2+2\lambda}{\lambda-1} + 2\lambda^2 + \lambda}{\lambda}\right) w(I) \\ &= \mu \cdot w(I) \end{aligned}$$

Case 2: Interval I is accepted through the algorithm's predictions rule. Notice that in this case, all conflicting intervals must have been accepted through the main rule, and not the predictions rule. Because of this, as we showed in case 1, for every $J \in I_s$, it holds that $\Phi(J) \leq \mu \cdot w(J)$. This helps us bound the amount of transfer charge to interval I.

$$\begin{aligned} \Phi(I) &\leq \mu \cdot w_c + (2\lambda + 1) \cdot w(I) \\ &\leq \mu \cdot w(I) + (2\lambda + 1) \cdot w(I) \\ &= (\mu + \delta) \cdot w(I) \end{aligned}$$

To summarize, we have shown that in the worst case, $\Phi(I) \leq (\mu + \delta) \cdot w(I)$ for every $I \in ALG$. This concludes the proof. □

We note that for $\lambda > \frac{2+\sqrt{5}}{\sqrt{5}-1} \approx 3.42$, the consistency of our algorithm is already better than $2\phi + 1$, and 22.15-robust. We have shown we can get consistency better than the online bound of `LR`, while maintaining bounded robustness. We believe further improvement on the bounds of `Revoke-Proportional` is possible, with an analysis that looks more closely at the dependence between direct and transfer charging.
One may also be able to further improve the algorithm by accepting

Figure 6: NASA-iPSC dataset.
(a) Unit & Irrevocable, (b) Unit & Revoking, (c) Proportional & Irrevocable, (d) Proportional & Revoking

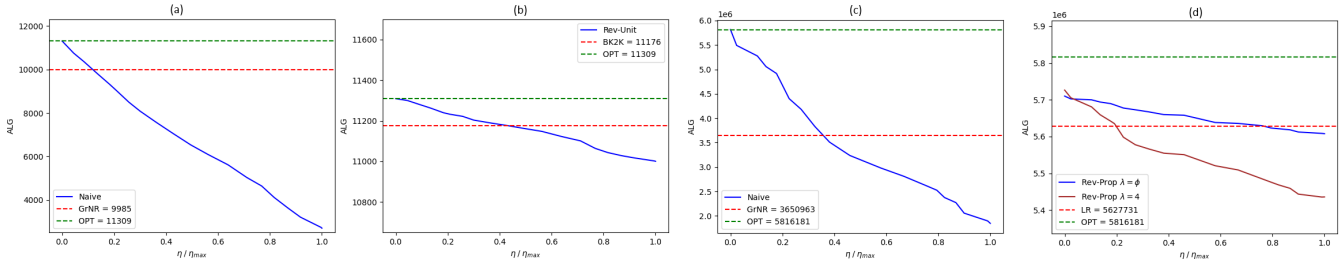
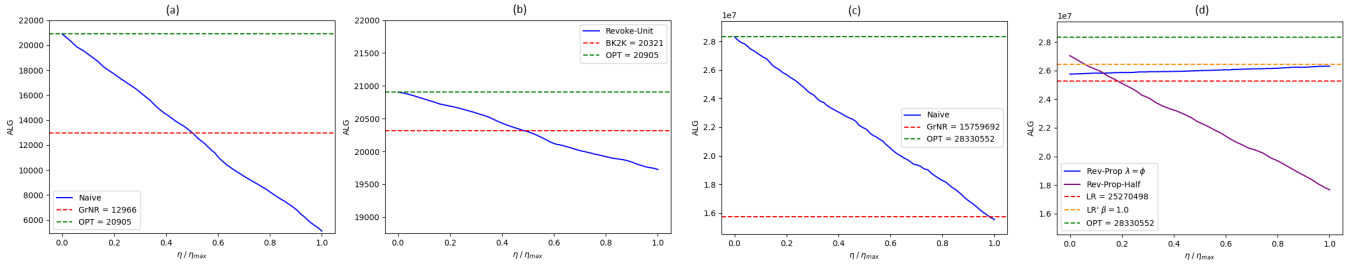


Figure 7: CTC-SP2 dataset.



an interval that is not as big as the sum of its conflicts, making the algorithm a-increasing with $a < 1$. This would relax the predictions rule further, and make the algorithm more prone to bad choices caused by misleading predictions. In our experiments, we briefly discuss one such algorithm, which we call `Revoke-Prop-Half`, and which can accept a supposedly optimal interval even if it is half as big as its conflicts.

5 EXPERIMENTAL RESULTS

We use real-world data from scheduling jobs on parallel machines[3] to test our algorithms. More information on the handling of these datasets can be found in a study by Feitelson et al. [17]. We focus on two datasets, `NASA-iPSC` (18,239 jobs) and `CTC-SP2` (77,222 jobs). As is usually the case, the performance of algorithms is much better than their worst-case bounds. For every algorithm we average its performance over random permutations of the input instance, for multiple error values. The y axis values for proportional weights are expressed in scientific notation. We note that algorithm `GrNR` refers to a greedy algorithm without revoking, a very natural algorithm to compare our `Naive` algorithm against. All other algorithms have been mentioned earlier in the paper. Our experimental results are in line with our intuition, with the predictions algorithms outperforming predictionless algorithms for some values of the error, even when they are not 1-consistent. Especially in the setting of revocable acceptances, even with half of the max possible error, our predictions algorithms perform just as well as their purely online counterparts. In the CTC dataset (figure 7) this is always the case, with the `Naive` algorithm outperforming `GrNR` for nearly all values of error. In the case of proportional weights with revoking in figure 6, it is noteworthy that the variant of `Revoke-Proportional` with $\lambda = 4$, outperforms the $\lambda = \phi$ variant for some small values of error, but its performance degrades faster. This further validates the notion that the bigger the λ, the more the algorithm follows the predictions.

We also have to address the seemingly abnormal behavior of algorithm `Revoke-Proportional` in figure 7(d). As the error increases, so does the performance of `Revoke-Proportional`, which is counterintuitive and dissimilar to the corresponding plot of figure 6. This is because of the underlying structure of the `CTC-SP2` dataset, on which *greedy* algorithms perform exceptionally well. We showcase this by having included algorithm `LR'` with $\beta = 1$, the algorithm that accepts a new interval if it is at least as big as everything it conflicts with. As the error increases, a larger number of intervals can be accepted through this clearly beneficial, relaxed predictions rule, which helps explain the improved performance. We also contrast this with algorithm `Rev-Prop-Half`, which uses a modified predictions rule, that can accept supposedly optimal intervals that are half the weight of their conflicts. This makes the algorithm more sensitive to the predictions, and its performance falls in line with what we would expect.

In conclusion, algorithms for interval selection can greatly benefit from utilizing imperfect predictions, and remain robust even in the presence of high error.

[3] https://www.cs.huji.ac.il/labs/parallel/workload/

ACKNOWLEDGMENTS

The author would like to thank Allan Borodin, Joan Boyar, and Kim Larsen for many helpful discussions, and for pointing out errors in earlier versions of this work.

REFERENCES

[1] Accessed: 2024-09. https://algorithms-with-predictions.github.io/.
[2] Matteo Almanza, Flavio Chierichetti, Silvio Lattanzi, Alessandro Panconesi, and Giuseppe Re. 2021. Online facility location with multiple advice. *Advances in neural information processing systems* 34 (2021), 4661–4673.
[3] Spyros Angelopoulos, Christoph Dürr, Shendan Jin, Shahin Kamali, and Marc Renault. 2024. Online computation with untrusted advice. *J. Comput. System Sci.* 144 (2024), 103545.
[4] Spyros Angelopoulos and Shahin Kamali. 2023. Contract scheduling with predictions. *Journal of Artificial Intelligence Research* 77 (2023), 395–426.
[5] Antonios Antoniadis, Joan Boyar, Marek Eliás, Lene Monrad Favrholdt, Ruben Hoeksma, Kim S Larsen, Adam Polak, and Bertrand Simon. 2023. Paging with succinct predictions. In *International Conference on Machine Learning*. PMLR, 952–968.
[6] Antonios Antoniadis, Hajo Broersma, and Yang Meng. 2024. Online Graph Coloring with Predictions. In *International Symposium on Combinatorial Optimization*. Springer, 289–302.
[7] Antonios Antoniadis, Christian Coester, Marek Eliáš, Adam Polak, and Bertrand Simon. 2023. Online metric algorithms with untrusted predictions. *ACM transactions on algorithms* 19, 2 (2023), 1–34.
[8] Antonios Antoniadis, Themis Gouleakis, Pieter Kleer, and Pavel Kolev. 2020. Secretary and online matching problems with machine learned advice. *Advances in Neural Information Processing Systems* 33 (2020), 7933–7944.
[9] Unnar Th Bachmann, Magnús M Halldórsson, and Hadas Shachnai. 2013. Online selection of intervals and t-intervals. *Information and Computation* 233 (2013), 1–11.
[10] Magnus Berg, Joan Boyar, Lene M Favrholdt, and Kim S Larsen. 2024. Complexity Classes for Online Problems with and without Predictions. *arXiv preprint arXiv:2406.18265* (2024).
[11] A. Borodin and R. El-Yaniv. Cambridge: Cambridge University Press, 1998. Online Computation and Competitive Analysis.
[12] Allan Borodin and Christodoulos Karavasilis. 2023. Any-order online interval selection. In *International Workshop on Approximation and Online Algorithms*. Springer, 175–189.
[13] Joan Boyar, Lene M Favrholdt, Shahin Kamali, and Kim S Larsen. 2023. Online interval scheduling with predictions. In *Algorithms and Data Structures Symposium*. Springer, 193–207.
[14] Joan Boyar, Lene M Favrholdt, Christian Kudahl, Kim S Larsen, and Jesper W Mikkelsen. 2017. Online algorithms with advice: A survey. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 1–34.
[15] Marek Elias, Haim Kaplan, Yishay Mansour, and Shay Moran. 2024. Learning-Augmented Algorithms with Explicit Predictors. *arXiv preprint arXiv:2403.07413* (2024).
[16] Yuval Emek, Magnús M Halldórsson, and Adi Rosén. 2016. Space-constrained interval selection. *ACM Transactions on Algorithms (TALG)* 12, 4 (2016), 1–32.
[17] Dror G Feitelson, Dan Tsafrir, and David Krakov. 2014. Experience with using the parallel workloads archive. *J. Parallel and Distrib. Comput.* 74, 10 (2014), 2967–2982.
[18] Virginie Gabrel. 1995. Scheduling jobs within time windows on identical parallel machines: New model and algorithms. *European Journal of Operational Research* 83, 2 (1995), 320–329.
[19] Juan A Garay, Inder S Gopal, Shay Kutten, Yishay Mansour, and Moti Yung. 1997. Efficient on-line call control algorithms. *Journal of Algorithms* 23, 1 (1997), 180–194.
[20] Elena Grigorescu, Young-San Lin, and Maoyuan Song. 2024. A Simple Learning-Augmented Algorithm for Online Packing with Concave Objectives. *arXiv preprint arXiv:2406.03574* (2024).
[21] Gupta, Lee, and Leung. 1979. An optimal solution for the channel-assignment problem. *IEEE Trans. Comput.* 100, 11 (1979), 807–810.
[22] Nicholas G Hall and Michael J Magazine. 1994. Maximizing the value of a space mission. *European journal of operational research* 78, 2 (1994), 224–241.
[23] Jon Kleinberg and Eva Tardos. 2006. *Algorithm design*. Pearson Education India.
[24] Antoon WJ Kolen, Jan Karel Lenstra, Christos H Papadimitriou, and Frits CR Spieksma. 2007. Interval scheduling: A survey. *Naval Research Logistics (NRL)* 54, 5 (2007), 530–543.
[25] Mikhail Y Kovalyov, Chi To Ng, and TC Edwin Cheng. 2007. Fixed interval scheduling: Models, applications, computational complexity and algorithms. *European journal of operational research* 178, 2 (2007), 331–342.
[26] Richard J Lipton and Andrew Tomkins. 1994. Online Interval Scheduling.. In *SODA*, Vol. 94. 302–311.
[27] Thodoris Lykouris and Sergei Vassilvitskii. 2021. Competitive caching with machine learned advice. *Journal of the ACM (JACM)* 68, 4 (2021), 1–25.
[28] Michael Mitzenmacher and Sergei Vassilvitskii. 2020. Algorithms with Predictions. In *Beyond the Worst-Case Analysis of Algorithms*, Tim Roughgarden (Ed.). Cambridge University Press, 646–662. https://doi.org/10.1017/9781108637435.037
[29] Serge Plotkin. 1995. Competitive routing of virtual circuits in ATM networks. *IEEE Journal on Selected Areas in Communications* 13, 6 (1995), 1128–1136.

[30] Manish Purohit, Zoya Svitkina, and Ravi Kumar. 2018. Improving online algorithms via ML predictions. *Advances in Neural Information Processing Systems* 31 (2018).

[31] Dhruv Rohatgi. 2020. Near-optimal bounds for online caching with machine learned advice. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 1834–1845.

[32] Andrew Tomkins. 1995. Lower bounds for two call control problems. *Information processing letters* 56, 3 (1995), 173–178.

[33] Gerhard J Woeginger. 1994. On-line scheduling of jobs with fixed start and end times. *Theoretical Computer Science* 130, 1 (1994), 5–16.