# LEARNING-AUGMENTED ONLINE ACKNOWLEDGMENT WITH WORST-CASE LATENCY GUARANTEES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We study the online acknowledgment problem under a maximum-latency objective, which captures tail latency and min–max fairness in systems where delayed responses may block downstream progress. An online algorithm observes packet arrivals over time and must decide when to issue acknowledgments, trading off acknowledgment overhead against the maximum delay among the packets covered by each acknowledgment.

We consider this problem in a learning-augmented setting, where the algorithm receives incremental (possibly erroneous) predictions of the next arrival time. We introduce a deterministic, parameterized algorithm, $\text{Augmented}_{\text{Alarm}}(\lambda)$, whose performance depends explicitly on the prediction error and a trust parameter $\lambda$. We give a tight, error-dependent characterization of its competitive ratio: it is $(1+\lambda)$-competitive when predictions are perfect (consistency), $(1+1/\lambda)$-competitive under arbitrary error (robustness), and exhibits smooth, graceful degradation as prediction accuracy deteriorates. We prove matching upper and lower bounds, showing that this consistency–robustness tradeoff is Pareto-optimal: no deterministic online algorithm can simultaneously achieve better consistency and robustness.

We further show that weaker prediction mechanisms, such as statistical information, are insufficient to achieve near-optimal performance, even when they are error-free.

Finally, we complement our theoretical results with experiments on synthetic and real-world packet traces, confirming the predicted smooth dependence on prediction error and the practical benefits of learning-augmented strategies.

## 1 INTRODUCTION

Many online systems must decide when to aggregate information and provide feedback in the absence of future knowledge. Delaying feedback can reduce overhead, but doing so too frequently inevitably increases latency. This fundamental tradeoff arises in a wide range of applications, including communication protocols (Dooly et al., 2001; Clark, 1982), distributed computation where tail latency dominates performance (Dean & Barroso, 2013), and online control systems where delayed feedback degrades stability and performance (Nilsson, 1998; Hespanha et al., 2007).

We study the online acknowledgment problem (Dooly et al., 2001), a simple yet expressive model capturing this tradeoff. An algorithm observes a sequence of packet arrivals over time and must decide when to issue acknowledgments. Each acknowledgment incurs a fixed cost, while postponing acknowledgments increases *latency*. A central question is how latency should be measured. While much prior work focuses on minimizing the *sum* of delays, aggregate objectives can mask extreme behavior, allowing a small number of requests to suffer large delays. We instead focus on the *maximum-latency* (or *max-delay*) objective, where the cost of an acknowledgment depends on the most delayed packet it covers. This objective is natural in settings where multiple arrivals jointly determine the completion of a downstream task or control decision: the task cannot proceed until all required packets have been acknowledged, and the delay is governed by the request that waited the longest. Thus, maximum latency directly models blocking behavior and captures a min–max notion of fairness: solutions that allow one request to incur excessive latency are penalized, regardless of how quickly others are served. This makes the objective useful for systems where delayed feedback is a bottleneck to progress (Dean & Barroso, 2013).



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

In the purely online setting, classical work established tight bounds for this problem: deterministic algorithms achieve a competitive ratio of 2, and this is optimal (Dooly et al., 2001). Modern online systems, however, increasingly rely on *predictions*, often learned from historical data, to guide real-time decision-making. This raises the question of whether such predictions can improve performance for maximum-latency objectives while retaining worst-case guarantees when they are inaccurate. We study this question in the framework of *learning-augmented online algorithms*, balancing *consistency* (performance under accurate predictions) with *robustness* (performance under arbitrary errors), without making probabilistic assumptions on the input or the prediction errors.

The prediction we use is deliberately local and incremental. Rather than assuming access to a predicted optimal solution or global future statistics, we assume that at each step the algorithm receives an estimate of the *next* arrival time. Equivalently, the algorithm may be viewed as issuing a *next-event query* and receiving a possibly noisy answer as the sequence unfolds. This interface matches time-series forecasting settings and avoids unrealistic "full-solution" predictions. Since the objective is governed by the worst delay, we measure prediction quality by the maximum deviation between predicted and actual arrival times.

**Contributions.** We study the online acknowledgment problem under the maximum-latency objective in a learning-augmented setting with incremental predictions of the next arrival time.

- We introduce a realistic prediction model for maximum-latency acknowledgment, where at each arrival, the algorithm receives a (possibly inaccurate) estimate of the next arrival time (Section 3).
- We revisit the classical *Alarm* family of algorithms for online acknowledgment (Dooly et al., 2001), analyzing their behavior under arrival-time predictions and showing that naively trusting predictions leads to unbounded worst-case latency (Section 4). This analysis motivates the design of $\text{Augmented}_{\text{Alarm}}(\lambda)$, a deterministic, parameterized algorithm that interpolates between trusting predictions and ignoring them (Section 5).
- We give an upper bound on the competitive ratio of $\text{Augmented}_{\text{Alarm}}(\lambda)$ as a function of the prediction error $\eta$ and the parameter $\lambda$ (Theorem 5.1). We also show that this upper bound is tight by proving a matching lower bound: for every value of $\eta$, there exist inputs on which $\text{Augmented}_{\text{Alarm}}(\lambda)$ attains this competitive ratio (Theorem 5.2).
- As a consequence, we establish a consistency–robustness tradeoff, proving that $\text{Augmented}_{\text{Alarm}}(\lambda)$ is $(1+\lambda)$-consistent and $(1+1/\lambda)$-robust (Corollary 5.3). We also prove that this tradeoff is Pareto-optimal: no deterministic online algorithm can simultaneously achieve better consistency and robustness (Theorem 5.4).
- Finally, we show that weaker prediction mechanisms are insufficient for achieving close-to-optimal solutions: even with perfect advice of sublinear size, no deterministic algorithm can achieve a competitive ratio strictly smaller than $1.1716$ (Theorem 6.1).
- We complement our theoretical results with an empirical evaluation on synthetic and real packet traces, demonstrating that $\text{Augmented}_{\text{Alarm}}(\lambda)$ exhibits smooth degradation as prediction error increases and outperforms classical baselines when predictions are moderately accurate (Section 7).

## 2 Related Work

**Online acknowledgment and latency objectives.** The online acknowledgment problem was introduced by Dooly et al. (2001) as a model for trading off the cost of issuing acknowledgments against the latency incurred by delaying them. The classical formulation considers two latency objectives: the *sum-delay* objective ($f_{\text{sum}}$), which aggregates the delays of all arrivals acknowledged together, and the *max-delay* (maximum-latency) objective ($f_{\max}$), which charges the maximum delay among them (our focus). For both objectives, Dooly et al. (2001) provided dynamic programs for the offline optimum (later improved to linear time by Noga et al. (2002)) and established tight competitive bounds in the online setting: deterministic algorithms achieve a competitive ratio of $2$, which is optimal. For the sum-delay objective, randomized competitive bounds are also known: the lower bound of $e/(e-1)$ (Seiden, 2000) is matched by the algorithm of Karlin et al. (2001) and by the primal-dual method of Buchbinder et al. (2007). Several variants of maximum-latency control have also been studied. Albers & Bals (2005) considered a different objective in which a single global maximum delay is added to the sum of acknowledgment costs, and derived tight (and in some

cases asymptotically tight) bounds for deterministic and randomized algorithms. Frederiksen et al. (2003) studied $f_{\max}$ under a minimum spacing constraint between consecutive acknowledgments and obtained optimal deterministic and randomized competitive ratios for that constrained setting.

**Min–max objectives and tail-latency control.** Maximum-type objectives are a standard approach to modeling tail behavior and worst-case responsiveness, and are closely connected to min–max notions of fairness; see surveys and frameworks such as Nace & Pióro (2009); Radunovic & Le Boudec (2007). In networking and resource allocation, min–max and lexicographic min–max criteria are used to prevent any single component from becoming a bottleneck (Georgiadis et al., 2001). Related worst-case formulations have also been explored in online settings, e.g., min–max paging, where the goal is to minimize the maximum number of faults incurred by any page (Chiplunkar et al., 2023). Our use of the $f_{\max}$ objective aligns with this perspective: performance is determined by the most delayed arrival, which captures tail-latency control and discourages solutions that "sacrifice" a small subset of arrivals to reduce average delay.

**Learning-augmented online algorithms.** A growing body of work studies how to incorporate machine-learned predictions into online algorithms while retaining worst-case guarantees; see surveys such as Mitzenmacher & Vassilvitskii (2020; 2022). The standard framework evaluates performance in terms of *consistency* under accurate predictions and *robustness* under arbitrary prediction errors, as initiated in seminal work (Purohit et al., 2018; Lykouris & Vassilvitskii, 2021). Learning-augmented techniques have been developed for many online problems, including scheduling, caching, matching, knapsack, and related combinatorial settings (e.g., Purohit et al. (2018); Lykouris & Vassilvitskii (2021); Rohatgi (2020); Wei (2020)).

For the acknowledgment problem, existing learning-augmented results focus on the sum-delay objective. Imreh & Németh (2011) proposed a parameter-learning approach for $f_{\text{sum}}$ and demonstrated empirical improvements without worst-case guarantees. In a stronger prediction model where a full predicted solution (predicted acknowledgment times) is given in advance, Bamas et al. (2020) designed a primal–dual algorithm that interpolates between following the prediction and ensuring bounded worst-case performance. Banerjee (2020) studied the acknowledgment under $f_{\text{sum}}$ under predictions of the offline-optimal schedule via a Hedge-style combination of strategies. Im et al. (2023) studied algorithms for $f_{\text{sum}}$ under predictions of arrival intensities and an error measure based on how the offline optimum changes between lower and upper envelope sequences.

**Our focus.** In contrast to the above, we study learning-augmented online acknowledgment under the maximum-latency objective $f_{\max}$, which directly targets tail-latency control and min–max responsiveness. We use an incremental prediction interface (next-arrival estimates revealed online) and analyze performance as an explicit function of prediction error. Optimizing $f_{\max}$ leads to algorithmic behavior and analyses that are fundamentally different from those under sum-based objectives. Our results provide tight error-dependent upper and lower bounds and are achieved using a deterministic, parameterized algorithm that avoids mixtures or expert-weighting schemes.

## 3 Model and Preliminaries

We consider the online acknowledgment problem, where packets (or requests) arrive over time and must be acknowledged by an online algorithm. Each packet $i$ is characterized solely by its arrival time $a_i \in \mathbb{R}_{\geq 0}$, and arrivals are revealed to the algorithm in increasing order of arrival times. Therefore, an input to the problem is an online sequence $\langle a_1, a_2, \ldots, a_n \rangle$ of arrival times, where $a_i < a_{i+1}$. The algorithm decides when to issue acknowledgments, without knowledge of future arrivals, or when the input ends. A solution consists of a sequence of acknowledgment times $t_1 < t_2 < \cdots < t_k$, where each packet arriving in the interval $(t_{j-1}, t_j]$ (with $t_0 = 0$) is acknowledged at time $t_j$.

Issuing an acknowledgment incurs two types of cost: a fixed *acknowledgment cost* $\rho \in (0, 1)$, capturing overhead, and a *latency cost*, reflecting the delay experienced by acknowledged packets.

We focus on the *maximum-latency* objective. For an acknowledgment at time $t_j$, the latency cost is the maximum delay among all packets acknowledged at that time, that is, $\max_{i: t_{j-1} < a_i \leq t_j} (t_j - a_i)$. The cost of the acknowledgment at time $t_j$ is therefore

$$c_j = \rho + (1 - \rho) \max_{i: t_{j-1} < a_i \leq t_j} (t_j - a_i).$$

The objective is to minimize the sum of the costs over all acknowledgments, i.e., to minimize $\sum_j c_j$. Note that $\rho$ is fixed and captures the relative cost of making an acknowledgment versus delaying it.

**Offline optimum and performance measures.** The *offline optimum* (OPT) is defined as the minimum possible total cost achievable by an algorithm with full knowledge of the entire arrival sequence in advance. We evaluate our algorithms using the standard notion of *competitive ratio*. An online algorithm ALG has competitive ratio at most $\gamma$ if, for all input sequences $\sigma$, we have $\text{ALG}(\sigma) \leq \gamma \cdot \text{OPT}(\sigma)$. This notion captures worst-case performance relative to OPT.

Following prior work on learning-augmented online algorithms, we evaluate performance using two complementary criteria: *consistency*, defined as the competitive ratio when predictions are perfect ($\eta = 0$), and *robustness*, defined as the worst-case competitive ratio under arbitrary prediction error.

An algorithm is called $(\kappa, \gamma)$-*competitive* if it has a competitive ratio at most $\kappa$ when $\eta = 0$, and a competitive ratio at most $\gamma$ for all values of $\eta$. This framework captures the trade-off between exploiting accurate predictions and maintaining strong worst-case guarantees.

**Learning-augmented model.** We study a learning-augmented variant of the problem in which the algorithm has access to predictions about future arrivals. Predictions are revealed *incrementally*: upon observing an arrival $a_j$, the algorithm receives a prediction $\hat{a}_{j+1}$ for the next arrival time $a_{j+1}$, with the convention that $\hat{a}_{j+1} = +\infty$ indicates that the input terminates after $a_j$.
This prediction interface can be viewed as a next-event estimation oracle that provides noisy information as the sequence unfolds, rather than a full predicted solution given in advance. Such incremental predictions are consistent with standard time-series forecasting settings.

**Prediction error.** We measure prediction quality using a worst-case notion of error. Let $\eta = \max_j |\hat{a}_j - a_j|$ denote the maximum deviation between predicted and actual arrival times. This error measure, corresponding to $\ell_\infty$ distance between actual and predicted arrival vectors, is natural for the max-latency objective: since each acknowledgment cost is determined by the largest delay in the corresponding phase, the relevant prediction error is the largest local deviation affecting the timing of acknowledgments. In contrast, a measure like $\ell_1$ error aggregates deviations over time and may obscure the distinction between many small errors and a single large error, even though the latter can dominate a max-type objective. The choice of $\ell_\infty$ also captures bounded interval predictions. If each prediction is given as an interval $[\hat{a}_j^{\min}, \hat{a}_j^{\max}]$ of width at most $2\eta$ containing the true next arrival, then using the midpoint as the predicted arrival time gives an $\ell_\infty$ error bound of $\eta$. Thus, bounded-width interval predictions can be handled within the same worst-case error framework.

**Notation.** We use $\sigma$ to denote an input arrival sequence and write $\text{ALG}(\sigma)$ and $\text{OPT}(\sigma)$ for the costs incurred by an online algorithm and the offline optimum denoted by OPT, respectively. For each arrival $a_j$ ($j \in [n]$), let $\text{ALG}(j)$ and $\text{OPT}(j)$ denote the cost incurred by $\text{Augmented}_{\text{Alarm}}(\lambda)$ and OPT, respectively, due to request $j$.
If an algorithm $\mathcal{A}$ sends an acknowledgment at time $a_j + q$ before the next arrival $a_{j+1}$, the cost incurred for the $j$-th arrival is $\rho + (1-\rho)q$. If no acknowledgment is sent before $a_{j+1}$, the incurred cost is $(1-\rho)(a_{j+1} - a_j)$. We denote by $\mathcal{A}(j)$ the cost incurred by algorithm $\mathcal{A}$ due to the $j$-th arrival. The total cost on an input sequence $\sigma$ is then $\mathcal{A}(\sigma) = \sum_{j=1}^{n} \mathcal{A}(j)$.

## 4 Alarm Family of Algorithms

The Alarm family of algorithms, introduced by Dooly et al. (2001), operates as follows. Upon observing a packet at time $a_j$, the algorithm sets an alarm for time $a_j + d_j$, where the choice of $d_j$ depends on the specific strategy. A decision is made either when the alarm triggers or when the next packet arrival, $a_{j+1}$, is observed, whichever occurs first. If $a_{j+1}$ arrives before the alarm triggers, the alarm is reset to $a_{j+1} + d_{j+1}$; otherwise, when the alarm triggers, the algorithm sends an acknowledgment for all pending packets.

A special case of the Alarm algorithm sets $d_j := \frac{\rho}{1-\rho}$, which yields a 2-competitive algorithm. This algorithm is precisely the classical algorithm of Dooly et al. (2001), which we refer to as Classic. It is known that no deterministic algorithm can achieve a competitive ratio better than 2 for the $f_{\max}$ objective; hence, Classic is optimal among deterministic algorithms without additional information.

Dooly et al. (2001) also studied the Alarm algorithm in the presence of *lookahead*, where the algorithm is given perfect knowledge of the next arrival time $a_{j+1}$ upon observing $a_j$. Under this assumption, they showed that a suitable member of the Alarm family is optimal.

In what follows, we revisit this lookahead-based algorithm in the more realistic *prediction* setting, where the algorithm is given a (possibly inaccurate) prediction $\hat{a}_{j+1}$ of the next arrival. We refer to this algorithm as Trust. Upon observing packet $a_j$ and prediction $\hat{a}_{j+1}$, Trust behaves as follows:

- If $\hat{a}_{j+1} - a_j > \frac{\rho}{1-\rho}$, it immediately sends an acknowledgment at time $a_j$;
- Otherwise, if $\hat{a}_{j+1} - a_j \leq \frac{\rho}{1-\rho}$, it defers the decision and waits for the next arrival.

**Lemma 4.1.** *(Dooly et al., 2001) The Trust algorithm is optimal (*1*-competitive) when predictions are perfect, i.e., when $\hat{a}_j = a_j$ for all $j$.*

Intuitively, perfect knowledge of the next arrival allows the algorithm to decide optimally between acknowledging immediately and waiting.

We now show that Trust is not robust when predictions (lookaheads) are inaccurate. In particular, blindly trusting predictions is fundamentally incompatible with worst-case latency guarantees, even on the simplest nontrivial inputs.

**Lemma 4.2.** *Trust has an unbounded competitive ratio when predictions are inaccurate.*

*Proof.* Consider an input consisting of two packets $\langle a_1 = 0, a_2 = L \rangle$, where $L$ is arbitrarily large. Suppose the prediction at time 0 gives $\hat{a}_2 = \epsilon$ for sufficiently small $\epsilon$. Trust chooses to wait for the next arrival before acknowledging, thereby incurring a cost of $L(1-\rho)$. OPT, however, acknowledges at times $a_1$ and $a_2$, and has cost $2\rho$. As $L \to \infty$, the competitive ratio of Trust becomes unbounded. □

One might hope to mitigate this behavior by modifying Trust so that it ignores predictions once they are observed to be incorrect. However, the following lemma shows that even such a variant still cannot guarantee a bounded competitive ratio. The proof follows directly from the definition of the Alarm family and is deferred to the Appendix.

**Lemma 4.3** (Appendix §A.1)**.** *A variant of Trust that discards predictions once they are detected to be inaccurate has a competitive ratio of at least $\frac{n\rho}{\rho + n\epsilon(1-\rho)}$.*

The above results demonstrate that fully trusting predictions is incompatible with robustness.

## 5 Augmented$_{\text{Alarm}}(\lambda)$: A Robust and Consistent Algorithm

We present Augmented$_{\text{Alarm}}(\lambda)$, a member of the Alarm family parameterized by a trust parameter $\lambda \in (0, 1)$. The parameter $\lambda$ is a *hyperparameter* that controls the level of trust in predictions: smaller values of $\lambda$ favor consistency (performance under accurate predictions), while larger values improve robustness. We treat $\lambda$ as a fixed design choice, selected based on prior knowledge of prediction quality or via offline calibration on historical data. This parameter interpolates between fully trusting predictions and ignoring them entirely. As $\lambda \to 0$, the algorithm behaves similarly to Trust; when $\lambda = 1$, it coincides with the classical algorithm Classic.

**Augmented$_{\text{Alarm}}(\lambda)$ Algorithm**

Upon the arrival of packet $a_j$, use the prediction $\hat{a}_{j+1}$ to choose between two strategies:

- **Short Alarm.** If the prediction suggests acknowledging immediately, i.e., $\hat{a}_{j+1} - a_j > \frac{\rho}{1-\rho}$, set an alarm for time $a_j + \lambda\frac{\rho}{1-\rho}$.
- **Long Alarm.** If the prediction suggests waiting, i.e., $\hat{a}_{j+1} - a_j \leq \frac{\rho}{1-\rho}$, set an alarm for time $a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$.

### 5.1 Competitive Analysis of Augmented$_{\text{Alarm}}(\lambda)$

We analyze the competitive ratio of Augmented$_{\text{Alarm}}(\lambda)$ as a function of both $\lambda$ and the prediction error $\eta$, and later show that this interpolation yields tight consistency–robustness trade-offs.

**Theorem 5.1.** *The competitive ratio of the Augmented$_{Alarm}(\lambda)$ algorithm as a function of the prediction error $\eta$ can be upper-bounded as follows:*

$$CR(\eta,\lambda) \le \begin{cases} \frac{(1+\lambda)\rho}{\rho-\eta(1-\rho)} & \text{if } \eta \le (1-\lambda)\frac{\rho}{1-\rho} \\ 1+\frac{1}{\lambda}, & \text{if } \eta > (1-\lambda)\frac{\rho}{1-\rho}. \end{cases}$$

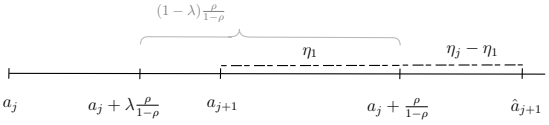
*Proof.* Fix an input sequence $\sigma = (a_1, a_2, \ldots, a_n)$. For each arrival $a_j$ ($j \in [n]$), let $A(j)$ and $\text{OPT}(j)$ denote the cost incurred by Augmented$_{\text{Alarm}}(\lambda)$ and OPT, respectively, due to request $j$.

If Augmented$_{\text{Alarm}}(\lambda)$ (resp. OPT) sends an acknowledgment at time $a_j + q$ before the next arrival $a_{j+1}$, then the incurred cost is $\rho + (1-\rho)q$. If no acknowledgment is sent before $a_{j+1}$, the incurred cost is $(1-\rho)(a_{j+1} - a_j)$. By definition, $\text{ALG}(\sigma) = \sum_{j=1}^{n} A(j)$ and $\text{OPT}(\sigma) = \sum_{j=1}^{n} \text{OPT}(j)$. Thus, to prove that Augmented$_{\text{Alarm}}(\lambda)$ has competitive ratio at most $c$, it suffices to show that $A(j) \le c \cdot \text{OPT}(j)$ for every $j \in [n]$.

For each $j$, define the prediction error $\eta_j = |\hat{a}_{j+1} - a_{j+1}|$ and recall that $\eta = \max_{j\in[n]} \eta_j$. We prove that for every $j$, $A(j) \le \frac{(1+\lambda)\rho}{\rho-\eta_j(1-\rho)}\,\text{OPT}(j)$ when $\eta_j \le (1-\lambda)\frac{\rho}{1-\rho}$, and $A(j) \le \left(1+\frac{1}{\lambda}\right)\text{OPT}(j)$ otherwise. The claimed competitive-ratio bound then follows immediately. We will use a case analysis to establish the proof. Recall that OPT sends an acknowledgment at time $a_j$ if $a_{j+1} - a_j \ge \frac{\rho}{1-\rho}$, where $\text{OPT}(j) = \rho$, and otherwise, waits till $a_{j+1}$, in which case its cost will be $(1-\rho)(a_{j+1} - a_j)$.

**Case (I): Short Alarm:** suppose $\hat{a}_{j+1} > a_j + \frac{\rho}{1-\rho}$, so Augmented$_{\text{Alarm}}(\lambda)$ plans a short alarm.

- *(Ia)* If $a_{j+1} \le a_j + \lambda\frac{\rho}{1-\rho}$, neither Augmented$_{\text{Alarm}}(\lambda)$ nor OPT sends an acknowledgment before $a_{j+1}$, and hence $A(j) = \text{OPT}(j)$.
- *(Ib)* If $a_{j+1} \ge a_j + \frac{\rho}{1-\rho}$, Augmented$_{\text{Alarm}}(\lambda)$ sends an acknowledgment at time $a_j + \lambda\frac{\rho}{1-\rho}$ and incurs cost $(1+\lambda)\rho$. Meanwhile, OPT acknowledges immediately and incurs cost $\rho$. Thus $A(j)/\text{OPT}(j) = (1+\lambda) \le \frac{(1+\lambda)\rho}{\rho-\eta(1-\rho)}$.
- *(Ic)* If $a_j + \lambda\frac{\rho}{1-\rho} < a_{j+1} < a_j + \frac{\rho}{1-\rho}$, Augmented$_{\text{Alarm}}(\lambda)$ again incurs cost $(1+\lambda)\rho$, while $\text{OPT}(j) = (1-\rho)(a_{j+1} - a_j)$. Since $a_{j+1} > a_j + \lambda\frac{\rho}{1-\rho}$, we have $\text{OPT}(j) \ge \lambda\rho$, yielding $A(j)/\text{OPT}(j) \le 1 + 1/\lambda$. This upper bound holds for any value of $\eta$; for smaller values of $\eta$, however, we can establish a better bound. Suppose $\eta_j \le (1-\lambda)\frac{\rho}{1-\rho}$. Given that $a_{j+1} < a_j + \frac{\rho}{1-\rho}$, we can write $a_{j+1} = a_j + \frac{\rho}{1-\rho} - \eta_1$ for some $\eta_1 \le (1-\lambda)\frac{\rho}{1-\rho}$ (see the figure below). Moreover, since $\hat{a}_{j+1} \ge a_j + \frac{\rho}{1-\rho}$, we have $\eta_j \ge \eta_1$. Therefore, we can write $\text{OPT}(j) = (1-\rho)(\frac{\rho}{1-\rho} - \eta_1) = \rho - \eta_1(1-\rho) \ge \rho - \eta_j(1-\rho)$. To conclude, when $\eta_j \le (1-\lambda)\frac{\rho}{1-\rho}$, we have $\frac{\text{ALG}(j)}{\text{OPT}(j)} \le \frac{(1+\lambda)\rho}{\rho-\eta_j(1-\rho)}$.

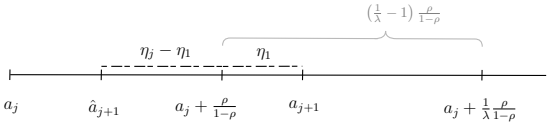



*Summary.* If Augmented$_{\text{Alarm}}(\lambda)$ plans a short alarm, then $\frac{\text{ALG}(j)}{\text{OPT}(j)} \le \frac{(1+\lambda)\rho}{\rho-\eta(1-\rho)}$ in Cases *(Ia)* and *(Ib)* for all prediction errors, and also in Case *(Ic)* when $\eta_j \le (1-\lambda)\frac{\rho}{1-\rho}$. For larger errors in Case *(Ic)*, the ratio is instead bounded by $\frac{\text{ALG}(j)}{\text{OPT}(j)} \le 1 + \frac{1}{\lambda}$.

**Case (II): Long Alarm:** Suppose $\hat{a}_{j+1} \le a_j + \frac{\rho}{1-\rho}$, so Augmented$_{\text{Alarm}}(\lambda)$ plans a long alarm.

- *(IIa)* If $a_{j+1} \le a_j + \frac{\rho}{1-\rho}$, neither algorithm sends an acknowledgment, so $A(j) = \text{OPT}(j)$.
- *(IIb)* If $a_{j+1} \ge a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$, then $\eta_j > (\frac{1}{\lambda} - 1)\frac{\rho}{1-\rho} > (1-\lambda)\frac{\rho}{1-\rho}$. In this case, Augmented$_{\text{Alarm}}(\lambda)$ acknowledges at $a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$ and incurs cost $\rho(1+1/\lambda)$, while $\text{OPT}(j) = \rho$, giving ratio $1 + 1/\lambda$.
- *(IIc)* If $a_j + \frac{\rho}{1-\rho} < a_{j+1} < a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$, then $\text{OPT}(j) = \rho$ and Augmented$_{\text{Alarm}}(\lambda)$ incurs cost at most $\rho(1+1/\lambda)$, yielding ratio at most $1 + 1/\lambda$. For smaller error values, we can obtain a better

upper bound for the ratio. Suppose $\eta \leq (1-\lambda)\frac{\rho}{1-\rho}$, which implies $\eta \leq (1/\lambda - 1)\frac{\rho}{1-\rho}$. Given that $a_{j+1} > a_j + \frac{\rho}{1-\rho}$, the delay of ALG$(j)$ can be written as $\frac{\rho}{1-\rho} + \eta_1$, for some $\eta_1 \in (0, (\frac{1}{\lambda} - 1)\frac{\rho}{1-\rho}]$ (see the figure below for illustration). Moreover, since $\hat{a}_{j+1} < a_j + \frac{\rho}{1-\rho}$, we have $\eta_j > \eta_1$. Hence, ALG$(j) = (\frac{\rho}{1-\rho} + \eta_1)(1-\rho) = \rho + \eta_1(1-\rho) \leq \rho + \eta_j(1-\rho)$. And the ratio would be $\frac{\text{ALG}(j)}{\text{OPT}(j)} = \frac{\rho+\eta_j(1-\rho)}{\rho} = 1 + \eta_j\frac{1-\rho}{\rho}$. Given that $\rho, \lambda \in [0,1]$, this ratio is at most $\frac{(1+\lambda)\rho}{\rho-\eta(1-\rho)}$.




*Summary.* To summarize, when Augmented$_{\text{Alarm}}(\lambda)$ plans a long alarm, in Case *(IIa)* we have $\frac{\text{ALG}(j)}{\text{OPT}(j)} = 1$ for any prediction error. In Case *(IIb)*, the prediction error satisfies $\eta_j > (1-\lambda)\frac{\rho}{1-\rho}$ and $\frac{\text{ALG}(j)}{\text{OPT}(j)} \leq 1 + \frac{1}{\lambda}$. Finally, in Case *(IIc)*, if $\eta_j \leq (1-\lambda)\frac{\rho}{1-\rho}$ then $\frac{\text{ALG}(j)}{\text{OPT}(j)} \leq \frac{(1+\lambda)\rho}{\rho-\eta(1-\rho)}$, whereas for larger error values, we again obtain $\frac{\text{ALG}(j)}{\text{OPT}(j)} \leq 1 + \frac{1}{\lambda}$. □

Next, we establish a matching lower bound on the competitive ratio of Augmented$_{\text{Alarm}}(\lambda)$, showing that the upper bound in Theorem 5.1 is tight; the full proof is deferred to the Appendix.

**Theorem 5.2** (Appendix §A.2)**.** *The competitive ratio of the Augmented$_{\text{Alarm}}(\lambda)$ algorithm as a function of the prediction error $\eta$ can be lower-bounded as follows:*

$$CR(\eta, \lambda) \geq \begin{cases} \frac{(1+\lambda)\rho}{\rho-\eta(1-\rho)} & \text{if } \eta \leq (1-\lambda)\frac{\rho}{1-\rho} \\ 1 + \frac{1}{\lambda}, & \text{if } \eta > (1-\lambda)\frac{\rho}{1-\rho}. \end{cases}$$

**Proof sketch.** The lower bound is established via adversarial input sequences tailored to different ranges of the prediction error $\eta$. For small error, the adversary constructs sequences in which Augmented$_{\text{Alarm}}(\lambda)$ is repeatedly induced to send premature acknowledgments, leading to a cost that scales with $(1+\lambda)\rho$ per arrival, while the offline optimum amortizes delay over a long sequence. For intermediate error values, the construction forces the algorithm into the short-alarm regime despite the existence of a cheap deferred acknowledgment, yielding a competitive ratio approaching $1 + 1/\lambda$. Finally, for large prediction error, the adversary exploits inaccurate predictions to trigger the long-alarm strategy while ensuring that the next arrival occurs just beyond the alarm time, leading to a tight lower bound of $1 + 1/\lambda$. Full details of the constructions and analysis are deferred to Appendix A.2.

**Consistency–Robustness Tradeoff.** Theorems 5.1 and 5.2 together establish an exact characterization of the competitive ratio of Augmented$_{\text{Alarm}}(\lambda)$ as a function of the prediction error $\eta$ and the parameter $\lambda$. In particular, when predictions are perfect (i.e., $\eta = 0$), Augmented$_{\text{Alarm}}(\lambda)$ achieves a competitive ratio of $1 + \lambda$, whereas for sufficiently large prediction error, its competitive ratio degrades to $1 + \frac{1}{\lambda}$. Consequently, Augmented$_{\text{Alarm}}(\lambda)$ is $(1+\lambda)$-consistent and $(1 + \frac{1}{\lambda})$-robust.

**Corollary 5.3.** *The Augmented$_{\text{Alarm}}(\lambda)$ algorithm is $(1 + \lambda,\ 1 + \frac{1}{\lambda})$-competitive.*

We show the tradeoff is optimal: no algorithm can simultaneously have better consistency and robustness. Therefore, Augmented$_{\text{Alarm}}(\lambda)$ is Pareto-optimal with respect to consistency and robustness.

**Theorem 5.4** (Appendix §A.3)**.** *Let $\lambda \in (0, 1)$. For any deterministic algorithm $B$, either its robustness is at least $1 + \frac{1}{\lambda}$, or its consistency strictly exceeds $1 + \lambda$.*

**Proof sketch.** We use inputs of length at most 2, with first arrival $a_1 = 0$ and prediction $\hat{a}_2 = +\infty$. Any deterministic algorithm $B$ acknowledges at some time $t$. If $t \leq \lambda\frac{\rho}{1-\rho}$, an adversary places $a_2 = t + \varepsilon$, forcing robustness at least $1 + \frac{1}{\lambda}$. If $t > \lambda\frac{\rho}{1-\rho}$, the adversary sets $a_2 = +\infty$, forcing consistency strictly above $1 + \lambda$. Thus no deterministic algorithm can achieve both simultaneously.

**Smoothness.** Augmented$_{\text{Alarm}}(\lambda)$ exhibits a clear form of *smoothness* with respect to the prediction error $\eta$: as the error increases, the competitive ratio deteriorates in a gradual and well-behaved manner rather than experiencing abrupt changes. This smooth dependence is visible in Figure 1,

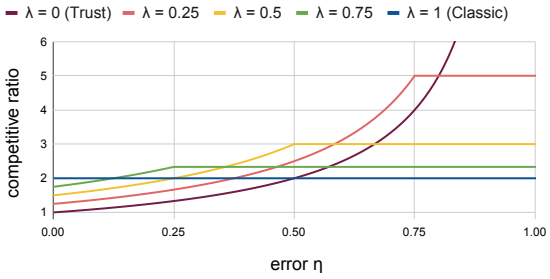



Figure 1: Competitive ratio of $\text{Augmented}_{\text{Alarm}}(\lambda)$ as a function of the prediction error $\eta$, for $\rho = 0.5$.

where the competitive ratio increases continuously as a function of $\eta$, smoothly interpolating between the consistency regime (small $\eta$) and the robustness regime (large $\eta$). Such graceful degradation is desirable in practice, as it ensures stability under moderately inaccurate predictions.

## 6 Limitations of Weaker Predictions

Our learning-augmented model assumes incremental predictions of the next arrival time, revealed as the input sequence unfolds. This choice occupies a natural middle ground between two extremes. On one end, prior work on the $f_{\text{sum}}$ objective has considered substantially stronger predictions, such as full knowledge of the optimal acknowledgment schedule or aggregate future arrival counts. While such predictions can yield strong guarantees, they are difficult to justify in practice and unlikely to be reliably learnable. On the other end of the spectrum, one may ask whether significantly weaker forms of prediction suffice; for instance, compact statistical summaries of the input (e.g., averages or variances of inter-arrival times), or predictions whose total information content grows sublinearly with the input length. To formalize this question, we turn to the classical *advice* model. Advice can be viewed as a form of perfect, error-free prediction whose sole constraint is its size, measured in bits. In particular, sublinear advice captures a broad class of compact predictions, including many PAC-learnable or statistical descriptions of the arrival sequence.

We show that such weak predictions are insufficient for near-optimal performance under the $f_{\max}$ objective, even when they are perfect. In particular, sublinear advice cannot achieve a competitive ratio arbitrarily close to 1. This separates compact or statistical predictions from richer forms of future information, such as full lookahead or arrival-time predictions of linear size.

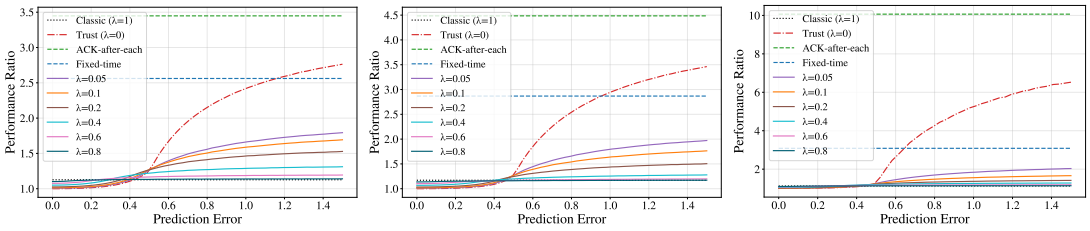
**Theorem 6.1** (Appendix §A.4)**.** *In the online acknowledgment problem under the $f_{max}$ objective, no deterministic algorithm with $o(n)$ bits of advice, where $n$ is the number of arrivals, can achieve a competitive ratio strictly smaller than* $1.1716$.

**Proof sketch.** The proof proceeds via a reduction from the classical *Binary String Guessing* problem, a canonical problem in advice complexity. In this problem, an online algorithm must predict a binary sequence revealed one bit at a time, incurring a unit cost for each incorrect guess. It is known that with $o(n)$ bits of advice, no algorithm can correctly predict more than a $1/2 + \varepsilon$ fraction of the bits for any constant $\varepsilon > 0$ (Böckenhauer et al., 2014). We reduce Binary String Guessing to the online acknowledgment problem under the $f_{\max}$ objective by constructing an input sequence composed of independent phases. Each phase contains either one or two arrivals, depending on the corresponding bit value, and forces the algorithm to choose between acknowledging immediately or waiting for a possible second arrival. This choice naturally encodes a guess for the bit value.

A similar limitation extends to natural noisy-advice models, where predictions are correct only with some probability. Such models also fail to overcome the fundamental barrier for the $f_{\max}$ objective. We defer the details to the Appendix §A.5.

## 7 Experiments

In this section, we empirically evaluate the performance of the proposed family of algorithms under varying levels of prediction error. Our goal is to empirically illustrate the qualitative behavior predicted by the theory and to examine how the algorithms perform across different levels of prediction error. We consider both synthetic workloads and real packet traces.




(a) Poisson-distributed arrivals. (b) Pareto-distributed arrivals. (c) MAWI dataset arrivals.

Figure 2: Performance ratio as a function of the prediction error $\eta$, across three arrival models: (a) synthetic sequences generated from a Poisson distribution, (b) synthetic sequences generated from a Pareto distribution, and (c) real-world sequences extracted from the MAWI traffic trace dataset.

**Synthetic data.** We evaluate the algorithms on two standard synthetic models of packet arrivals. First, we generate Poisson arrival sequences by sampling inter-arrival times from an exponential distribution with scale parameter $0.3$, and taking cumulative sums to obtain packet arrival times. This model has historically been used to capture TCP traffic and serves as a classical baseline. Second, to better reflect burstiness and heavy-tailed behavior observed in real network traffic (Paxson & Floyd, 1995), we generate arrivals using a Pareto distribution with shape parameter $\alpha = 3$ and scale $0.5$, following prior work (Bamas et al., 2020). The Pareto model produces frequent short inter-arrivals interspersed with occasional long gaps, creating instances with both bursts and idle periods.

**Real-world data.** We additionally evaluate our algorithms on real packet traces from the MAWI Working Group Traffic Archive (Cho et al., 2000), a widely used repository of Internet backbone traffic. Each trace consists of 15 minutes of packet-level data collected on a trans-Pacific link between Japan and the United States. To enable comparison with synthetic data, arrival timestamps are rescaled so that the mean inter-arrival time is $0.2$.

**Implementation details.** We set $\rho = 0.5$, making the cost of delaying a packet by one time unit equal to the cost of sending an acknowledgment. Prediction error is introduced by perturbing each arrival time with additive noise drawn uniformly at random from $[-\eta, \eta]$, where $\eta$ controls the error magnitude. We vary $\eta$ from $0$ to $1.5$. Each experiment is repeated over $1000$ independent trials, each with $1000$ arrivals. We report the average performance ratio, defined as the average of the ratio of the costs of ALG and OPT across 1000 trials. Appendix §B reports additional plots with shaded min–max ranges. The supplementary material contains an anonymized code repository.

**Algorithms.** We evaluate $\text{Augmented}_{\text{Alarm}}(\lambda)$ for $\lambda \in \{0.05, 0.1, 0.2, 0.4, 0.6, 0.8\}$. For comparison, we include the two extremal algorithms: Trust (where $\lambda \to 0$) and Classic ($\lambda = 1$). We further compare against two simple baselines: *ACK-after-each*, which acknowledges every packet immediately (no delay cost), and *Fixed-time*, which sends acknowledgments after a fixed waiting time (equal to the known mean inter-arrival time), provided there is at least one pending packet.

**Experimental analysis.** Figure 2 reports the average performance ratio as a function of the prediction error $\eta$. Across all benchmarks, we observe highly consistent trends: algorithms with smaller $\lambda$ excel under accurate predictions but degrade quickly as the error increases, while moderate values of $\lambda$ provide stable performance across a wide range of prediction errors. When predictions are perfect ($\eta = 0$), following the prediction is optimal: Trust achieves ratio 1, and smaller values of $\lambda$ consistently yield better performance. As prediction error increases, however, algorithms that rely heavily on predictions degrade more sharply, reflecting the lack of robustness predicted by theory. Across all datasets, moderate values of $\lambda$ achieve a favorable tradeoff: they significantly improve performance in the low-error regime while avoiding the catastrophic blow-up observed for Trust under larger errors. This behavior closely mirrors the theoretical bounds established in Section 5.

Figure 2a and Figure 2b illustrate performance across synthetic arrival models. Across both distributions, $\text{Augmented}_{\text{Alarm}}(\lambda)$ exhibits smooth degradation as prediction error increases, with smaller values of $\lambda$ performing well for accurate predictions and larger values providing improved robustness. Figure 2c shows similar trends on real-world data. In all cases, even small values of $\lambda > 0$ consistently outperform purely online baselines (in particular, Classic) when the prediction error $\eta$ is not too large.

### AI USE STATEMENT

In this work, we used generative AI tools (OpenAI (2026) and Anthropic (2026)) to assist with grammar and language editing across all sections, based on a complete and polished draft written by the authors. Anthropic Claude was also used as a coding assistant to support refactoring of the code used to obtain the empirical evidence and plot the graphs. We have not used generative AI tools for problem formulation, theorems, proofs, algorithm or experiment design, implementation of methods, data generation or cleaning, or interpretation of results. We reviewed all AI-assisted work. We checked text edits against the original draft and verified the refactored code to reproduce the original results with fixed seeds. We take responsibility for the final content of this work.

### REPRODUCIBILITY STATEMENT

All theoretical results are stated with their assumptions in Section 3; complete proofs appear in Sections 4–6 and Appendix A. The experimental setup (data sources, noise model, parameters, baselines, trials) is in Section 7, with additional details in Appendix B. Code for all algorithms, the offline optimum, data generation, and plots is in the supplementary material. MAWI traces (Cho et al., 2000) are publicly available.

## REFERENCES

Susanne Albers and Helge Bals. Dynamic TCP acknowledgment: Penalizing long delays. *SIAM Journal on Discrete Mathematics*, 19(4):938–951, 2005.

Anthropic. Claude, 2026. URL `https://claude.ai/`. Accessed: 2026-05-22.

Etienne Bamas, Andreas Maggiori, and Ola Svensson. The primal-dual method for learning augmented algorithms. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:20083–20094, 2020.

Shom Banerjee. Improving online rent-or-buy algorithms with sequential decision making and ML predictions. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:21072–21080, 2020.

Hans-Joachim Böckenhauer, Juraj Hromkovič, Dennis Komm, Sacha Krug, Jasmin Smula, and Andreas Sprock. The string guessing problem as a method to prove lower bounds on the advice complexity. *Theoretical Computer Science*, 554:95–108, 2014.

Niv Buchbinder, Kamal Jain, and Joseph Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *European Symposium on Algorithms*, pp. 253–264. Springer, 2007.

Ashish Chiplunkar, Monika Henzinger, Sagar Sudhir Kale, and Maximilian Vötsch. Online min-max paging. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1545–1565, 2023.

Kenjiro Cho, Koushirou Mitsuya, and Akira Kato. Traffic data repository at the WIDE project. In *2000 USENIX Annual Technical Conference (USENIX ATC 00)*, 2000.

David D Clark. Window and acknowledgement strategy in TCP. Technical Report RFC 813, MIT Laboratory for Computer Science, 1982.

Jeffrey Dean and Luiz André Barroso. The tail at scale. *Communications of the ACM*, 56(2):74–80, 2013.

Daniel R. Dooly, Sally A. Goldman, and Stephen D. Scott. Online analysis of the TCP acknowledgment delay problem. *Journal of ACM*, 48(2):243–273, 2001.

Jens S Frederiksen, Kim S Larsen, John Noga, and Patchrawat Uthaisombut. Dynamic TCP acknowledgment in the LogP model. *Journal of Algorithms*, 48(2):407–428, 2003.

Leonidas Georgiadis, Panos Georgatsos, Kostas Floros, and Stelios Sartzetakis. Lexicographically optimal balanced networks. In *IEEE Conference on Computer Communications (INFOCOM)*, volume 2, pp. 689–698, 2001.

Joo P Hespanha, Payam Naghshtabrizi, and Yonggang Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162, 2007.

Sungjin Im, Benjamin Moseley, Chenyang Xu, and Ruilong Zhang. Online dynamic acknowledgement with learned predictions. In *IEEE Conference on Computer Communications (INFOCOM)*, pp. 1–10, 2023.

Csanád Imreh and Tamás Németh. Parameter learning algorithm for the online data acknowledgment problem. *Optimization Methods and Software*, 26(3):397–404, 2011.

Anna R Karlin, Claire Kenyon, and Dana Randall. Dynamic TCP acknowledgement and other stories about $e/(e-1)$. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing (STOC)*, pp. 502–509, 2001.

Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *Journal of the ACM*, 68(4):1–25, 2021.

M. Mitzenmacher and S. Vassilvitskii. Algorithms with predictions. In *Beyond the Worst-Case Analysis of Algorithms*, pp. 646–662. Cambridge University Press, 2020.

Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with predictions. *Communications of the ACM*, 65(7):33–35, 2022.

Dritan Nace and Michal Pióro. Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial. *IEEE Communications surveys & tutorials*, 10(4):5–17, 2009.

Johan Nilsson. Real-time control systems with delays. *PhD Thesis TFRT-1049*, 1998.

John Noga, Steve Seiden, and Gerhard J Woeginger. A faster off-line algorithm for the TCP acknowledgement problem. *Information Processing Letters*, 81(2):71–73, 2002.

OpenAI. ChatGPT, 2026. URL `https://chat.openai.com/`. Accessed: 2026-05-22.

Vern Paxson and Sally Floyd. Wide area traffic: the failure of Poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, 1995.

Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ML predictions. *Advances in Neural Information Processing Systems (NeurIPS)*, 31:9661–9670, 2018.

Bozidar Radunovic and Jean-Yves Le Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Transactions on Networking*, 15(5):1073–1083, 2007.

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

Steven S Seiden. A guessing game and randomized online algorithms. In *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing (STOC)*, pp. 592–601, 2000.

Alexander Wei. Better and Simpler Learning-Augmented Online Caching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, volume 176, pp. 60:1–60:17, 2020.

# A Omitted Proofs

## A.1 Proof of Lemma 4.3

*Proof.* Consider an instance with $n$ arrivals spaced $\epsilon$ apart, i.e., $a_{j+1} - a_j = \epsilon$ for all $j$. Suppose that at each arrival, the prediction overestimates the next arrival time so that $\hat{a}_{j+1} - a_j > \frac{\rho}{1-\rho}$. Then Trust sends an acknowledgment after every arrival, incurring cost $\rho n$. The optimal solution sends a single acknowledgment after the final arrival, incurring cost $\rho + n\epsilon(1-\rho)$. The stated bound follows, and becomes unbounded for large $n$ and small $\epsilon$. □

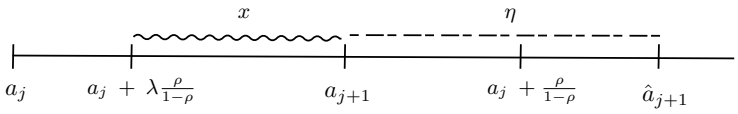
## A.2 Proof of Theorem 5.2

*Proof.* We construct three adversarial input sequences, each establishing the claimed lower bound on the competitive ratio of $\text{Augmented}_{\text{Alarm}}(\lambda)$ for a different range of prediction errors $\eta$.

- Let $\eta$ be any arbitrary value of at most $(1-\lambda)\frac{\rho}{1-\rho}$. Consider the input $\sigma_1 = \langle a_1, \ldots, a_n \rangle$, where $a_j = j(\lambda\frac{\rho}{1-\rho} + x)$ with $x = (1-\lambda)\frac{\rho}{1-\rho} - (\eta - \delta)$ for some sufficiently small $\delta$ (e.g., $\delta \leq 1/n^2$). Since $\eta$ satisfies the stated upper bound, $x$ is positive. One possible solution for this input is to send a single acknowledgment at the end. Hence,

$$\begin{aligned}
\text{OPT}(\sigma_1) &\leq n(1-\rho)(a_{j+1} - a_j) + \rho \\
&= n(1-\rho)(\lambda\frac{\rho}{1-\rho} + x) + \rho \\
&= n(1-\rho)(\lambda\frac{\rho}{1-\rho} + (1-\lambda)\frac{\rho}{1-\rho} - (\eta - \delta)) + \rho \\
&= n(1-\rho)(\frac{\rho}{1-\rho} - (\eta - \delta)) + \rho \\
&= n\rho - n(1-\rho)(\eta - \delta) + \rho
\end{aligned}$$

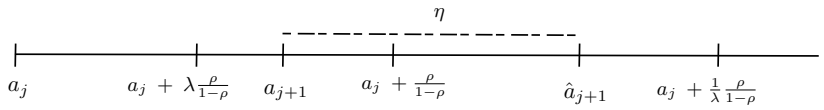
  Now, suppose the prediction specifies $\hat{a}_j = a_j + \eta$ for any $j \in [n]$; so the prediction error is exactly $\eta$. Given that $\hat{a}_{j+1} - a_j = \lambda\frac{\rho}{1-\rho} + x + \eta = \lambda\frac{\rho}{1-\rho} + ((1-\lambda)\frac{\rho}{1-\rho} - \eta + \delta) + \eta = \frac{\rho}{1-\rho} + \delta > \frac{\rho}{1-\rho}$, $\text{Augmented}_{\text{Alarm}}(\lambda)$ waits for a time of $\lambda(\frac{\rho}{1-\rho})$ before making an acknowledgment (it always enters the short-alarm case). Moreover, since $a_{j+1} = a_j + \lambda\frac{\rho}{1-\rho} + x > a_j + \lambda\frac{\rho}{1-\rho}$, $\text{Augmented}_{\text{Alarm}}(\lambda)$ makes an acknowledgment at time $a_j + \lambda\frac{\rho}{1-\rho}$, and its cost for each acknowledgment would be $\rho + (1-\rho)\lambda\frac{\rho}{1-\rho} = (1+\lambda)\rho$. This sums up to $(1+\lambda)n\rho$ for all $n$ acknowledgments, and thus the ratio between the cost of $\text{Augmented}_{\text{Alarm}}(\lambda)$ and OPT for $\sigma_1$ is $\frac{(1+\lambda)n\rho}{n\rho - n(1-\rho)(\eta-\delta)+\rho} \simeq \frac{(1+\lambda)\rho}{\rho - \eta(1-\rho)}$ for arbitrarily large $n$. The figure below illustrates this case.




- Let $\eta$ be any arbitrary value such that $(1-\lambda)\frac{\rho}{1-\rho} < \eta < (\frac{1}{\lambda} - 1)\frac{\rho}{1-\rho}$. Consider the input $\sigma_2 = \langle a_1, \ldots, a_n \rangle$, where $a_j = (\lambda\frac{\rho}{1-\rho} + \delta)j$ for sufficiently small $\delta$ (e.g., $\delta \leq 1/n^2$). Again, acknowledging once at the end yields

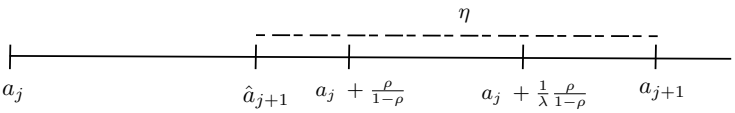
$$\begin{aligned}
\text{OPT}(\sigma_2) &\leq n(1-\rho)(a_{j+1} - a_j) + \rho \\
&= n(1-\rho)(\lambda\frac{\rho}{1-\rho} + \delta) + \rho \\
&= n\lambda\rho + n\delta(1-\rho) + \rho
\end{aligned}$$

Assume the prediction is $\hat{a}_j = a_j + \eta$ for all $j$, so the error is exactly $\eta$. Given that $\hat{a}_{j+1} - a_j = a_{j+1} + \eta - a_j = \lambda\frac{\rho}{1-\rho} + \delta + \eta > \lambda\frac{\rho}{1-\rho} + \delta + (1-\lambda)\frac{\rho}{1-\rho} > \frac{\rho}{1-\rho}$, Augmented$_{\text{Alarm}}(\lambda)$ waits for a time of $\lambda(\frac{\rho}{1-\rho})$ before making an acknowledgment (again, it always chooses the short-alarm strategy). Moreover, since $a_{j+1} = a_j + \lambda\frac{\rho}{1-\rho} + \delta > a_j + \lambda\frac{\rho}{1-\rho}$, Augmented$_{\text{Alarm}}(\lambda)$ makes an acknowledgment at time $a_j + \lambda\frac{\rho}{1-\rho}$, and its cost for each acknowledgment would be $\rho + (1-\rho)\lambda\frac{\rho}{1-\rho} = (1+\lambda)\rho$. This sums up to $(1+\lambda)n\rho$ for all $n$ acknowledgments, and thus the ratio between the cost of Augmented$_{\text{Alarm}}(\lambda)$ and OPT for $\sigma_2$ is $\frac{(1+\lambda)n\rho}{n\lambda\rho + n(1-\rho)(\delta)+\rho} \simeq \frac{1}{\lambda} + 1$ for arbitrarily large $n$. The figure below illustrates this case.




- Let $\eta$ be any arbitrary value larger than $(\frac{1}{\lambda} - 1)\frac{\rho}{1-\rho}$, that is $\eta = (\frac{1}{\lambda} - 1)\frac{\rho}{1-\rho} + \delta$ for some positive $\delta$. Consider the input arrival times $\sigma_3 = \langle a_1, \ldots, a_n \rangle$, where $a_j = j(\frac{\rho}{1-\rho} - \epsilon + \eta)$ for sufficiently small values of $\epsilon$ (say $\epsilon = \delta/n^2$). One solution for this input is to acknowledge every packet separately; thus OPT$(\sigma_3) \leq \rho n$. Now, suppose the prediction specifies that, for any $j \in [n]$, $\hat{a}_j = a_j - \eta$; thus, the prediction error is exactly $\eta$. Given that $\hat{a}_{j+1} - a_j = a_{j+1} - \eta - a_j = \frac{\rho}{1-\rho} - \epsilon < \frac{\rho}{1-\rho}$ Augmented$_{\text{Alarm}}(\lambda)$ chooses the long-alarm strategy, setting the alarm for $a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$. Moreover, since $\eta = (\frac{1}{\lambda} - 1)\frac{\rho}{1-\rho} + \delta$, we have $a_{j+1} = a_j + \frac{\rho}{1-\rho} - \epsilon + \eta > a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$. Therefore, Augmented$_{\text{Alarm}}(\lambda)$ makes an acknowledgment at time $a_j + \frac{1}{\lambda}\frac{\rho}{1-\rho}$, and its cost for each acknowledgment would be $\rho + (1-\rho)\frac{1}{\lambda}\frac{\rho}{1-\rho} = (1 + \frac{1}{\lambda})\rho$.

  This sums up to $(1 + \frac{1}{\lambda})n\rho$ for all $n$ acknowledgments, and thus the ratio between the cost of Augmented$_{\text{Alarm}}(\lambda)$ and OPT for $\sigma_3$ is $1 + \frac{1}{\lambda}$. The figure below provides an illustration.




□

### A.3 Proof of Theorem 5.4

*Proof.* We prove the claim using simple inputs of length at most 2, showing that the tradeoff already manifests in the most basic nontrivial setting. Let the first arrival occur at time $a_1 = 0$, and suppose the prediction specifies $\hat{a}_2 = +\infty$, indicating that no further arrivals are expected. Consider the behavior of algorithm $B$ upon observing $a_1$.

An adversary determines the second arrival time based on the moment $t$ at which $B$ issues an acknowledgment.

- Suppose that $B$ sends an acknowledgment at time $t \leq \lambda\frac{\rho}{1-\rho}$. The adversary then releases a second packet at time $a_2 = t + \epsilon$, for an arbitrarily small $\epsilon > 0$. In this case, $B$ incurs cost $\rho + (1-\rho)t$, while the optimal algorithm waits until $a_2$ and incurs cost $(1-\rho)(t+\epsilon)$. The resulting competitive ratio is $\frac{\rho+(1-\rho)t}{(1-\rho)(t+\epsilon)}$, which is decreasing in $t$ and is therefore minimized at $t = \lambda\frac{\rho}{1-\rho}$. Taking the limit as $\epsilon \to 0$, the ratio converges to $1 + \frac{1}{\lambda}$. Since the prediction error in this construction can be made arbitrarily large, we conclude that in this case the robustness of $B$ is at least $1 + \frac{1}{\lambda}$.
- Otherwise, suppose that $B$ sends an acknowledgment at time $t > \lambda\frac{\rho}{1-\rho}$. The adversary sets $a_2 = \hat{a}_2 = +\infty$, so that the input consists of a single arrival and the prediction is perfect (i.e., $\eta = 0$). Algorithm $B$ then incurs cost $\rho + (1-\rho)t > (1+\lambda)\rho$, while the optimal algorithm acknowledges immediately and incurs cost $\rho$. Thus, the consistency of $B$ exceeds $1 + \lambda$.

Therefore, $B$ must either have robustness at least $1 + \frac{1}{\lambda}$ or consistency larger than $1 + \lambda$. □

### A.4 Proof of Theorem 6.1

*Proof.* We begin by recalling the *Binary String Guessing problem*, a canonical online problem commonly used to prove advice lower bounds. In Binary String Guessing, an online algorithm is given a binary sequence $S = (S[1], \ldots, S[m])$ one bit at a time. Before seeing $S[i]$, the algorithm must guess its value. The cost is the total number of incorrect guesses. The algorithm may receive a limited number of advice bits that depend on the entire sequence in advance. A fundamental result (Böckenhauer et al., 2014) shows that with $o(m)$ bits of advice, no algorithm can correctly guess more than a $1/2 + \varepsilon$ fraction of the bits in an input binary string with an equal number of 0's and 1's, for any constant $\varepsilon > 0$.

We prove the theorem via a reduction from the Binary String Guessing problem. Let $S$ be an online sequence of $m$ bits, revealed one by one, forming an input to Binary String Guessing with an equal number of 0's and 1's. From $S$, we construct an instance $\sigma_S$ of the online acknowledgment problem consisting of $m$ phases.

The $i$th phase begins with a packet arriving at time $i \cdot C$, where $C$ is a sufficiently large constant (e.g., $C = m\rho$), ensuring that acknowledgments are never deferred across phases. If $S[i] = 1$, the phase contains no further arrivals. If $S[i] = 0$, a second packet arrives at time $i \cdot C + x$, where $x < \frac{\rho}{1-\rho}$ is a parameter specified below.

Intuitively, upon observing the first packet of each phase, the algorithm must decide whether to acknowledge immediately or to wait for a possible second arrival. This decision corresponds to guessing the value of $S[i]$: acknowledging before time $i \cdot C + x$ encodes a guess of 1, while waiting encodes a guess of 0.

Depending on the true bit value and the algorithm's guess, the incurred cost in each phase is summarized below:

| bit | cost of correct guess | cost of incorrect guess |
|---|---|---|
| 0 | $(1-\rho)x + \rho$ | $\geq 2\rho$ |
| 1 | $\rho$ | $\geq (1-\rho)x + \rho$ |

We choose $x = \frac{(\sqrt{2}-1)\rho}{1-\rho}$. With this choice, the ratio between the cost of an incorrect guess and a correct guess is $\sqrt{2}$ for both phase types: $\frac{2\rho}{(1-\rho)x+\rho} = \frac{(1-\rho)x+\rho}{\rho} = \sqrt{2}$. Since incorrect guesses are equally costly in relative terms, the most favorable scenario for the algorithm arises when it mispredicts phases of type 1, as they contribute less to the total cost.

It is known that with $o(m)$ bits of advice, no algorithm for Binary String Guessing can predict more than a $1/2 + \epsilon$ fraction of the bits correctly, for any constant $\epsilon > 0$ (Böckenhauer et al., 2014). Thus, even with perfect advice of sublinear size, the algorithm must incur a linear number of incorrect guesses.

The cost of an optimal algorithm that guesses all bits correctly satisfies

$$\text{OPT} \leq \tfrac{m}{2}\big((1-\rho)x + \rho\big) + \tfrac{m}{2}\rho = \tfrac{\sqrt{2}+1}{2} m\rho,$$

whereas any algorithm incurs a cost of at least

$$\text{ALG} \geq m\big((1-\rho)x + \rho\big) = \sqrt{2}\, m\rho.$$

The resulting competitive ratio is therefore

$$\text{CR} \geq \frac{\sqrt{2}m\rho}{(\sqrt{2}+1)m\rho/2} = \frac{2\sqrt{2}}{\sqrt{2}+1} \approx 1.1716,$$

which completes the proof. □

### A.5 Noisy Advice Model

We consider a noisy variant of the next-arrival prediction model studied in the main text. In this model, each phase begins with an arrival at time $T$, and the algorithm receives a prediction for the

next arrival time. The prediction is correct with probability $q > 1/2$, independently across phases (packets).

More precisely, each phase has one of two types. In a type-0 phase, a second arrival occurs at time $T + x$, where $x$ is appropriately chosen. In a type-1 phase, no second arrival occurs within the phase; equivalently, the next arrival occurs only after the phase separation time $T + C$, where $C$ is chosen sufficiently large. Thus, the correct prediction is $T + x$ in a type-0 phase and $T + C$ in a type-1 phase. With probability $q$, the algorithm receives the correct prediction, and with probability $1 - q$, it receives the prediction corresponding to the other phase type.

**Theorem A.1.** *Consider the online acknowledgment problem under the $f_{\max}$ objective in a noisy next-arrival prediction model, where each prediction is correct with probability $q > 1/2$, independently across phases. Then every deterministic algorithm has expected cost at least $\frac{2+\sqrt{2}-q}{\sqrt{2}+1}$ times the offline optimum, where the expectation is over the prediction noise.*

*Proof.* Consider an input formed by phases, where each phase begins with an arrival at time $T$ and is separated from the next phase by a sufficiently large constant $C$, so that carrying pending packets across phase boundaries only increases the cost and cannot improve the competitive ratio.

There are two phase types. In a type-0 phase, a second packet arrives at time $T + x$. In a type-1 phase, no second packet arrives within the phase; equivalently, the next arrival occurs only after time $T + C$. We set $x = \frac{(\sqrt{2}-1)\rho}{1-\rho}$. A phase is of type 0 with probability $p = \sqrt{2} - 1$, and of type 1 with probability $1 - p = 2 - \sqrt{2}$.

The prediction given at the beginning of the phase is a noisy next-arrival prediction. If the phase is of type 0, the correct prediction is $T + x$; if the phase is of type 1, the correct prediction is $T + C$. With probability $q$, the algorithm receives the correct prediction, and with probability $1 - q$, it receives the prediction corresponding to the other phase type. The noise is independent across phases.

We first compute the optimal costs in each phase. Since $x < \frac{\rho}{1-\rho}$, the offline optimum in a type-0 phase waits until the second arrival and acknowledges both packets together. Hence $\mathrm{OPT}_0 = \rho + (1 - \rho)x = \rho + (\sqrt{2} - 1)\rho = \sqrt{2}\rho$. In a type-1 phase, the offline optimum acknowledges the single packet immediately, so $\mathrm{OPT}_1 = \rho$.

Upon seeing the first arrival and the prediction, a deterministic online algorithm must choose whether to acknowledge before time $T + x$ or to wait at least until time $T + x$. Acknowledging before $T + x$ is the correct decision in a type-1 phase, while waiting until $T + x$ is the correct decision in a type-0 phase. If the algorithm makes the wrong decision in a type-0 phase, then it acknowledges the first packet before the second packet arrives, and the second packet must be acknowledged separately. Thus, its cost is at least $2\rho$, and the ratio in that phase is at least $\frac{2\rho}{\sqrt{2}\rho} = \sqrt{2}$. If the algorithm makes the wrong decision in a type-1 phase, then it waits until at least time $T + x$ before acknowledging the only packet. Its cost is at least $\rho + (1 - \rho)x = \sqrt{2}\rho$, while the offline optimum pays $\rho$. Hence, the ratio is again at least $\sqrt{2}$. Thus, in every phase, a correct decision has a ratio of at least 1, and an incorrect decision has a ratio at least $\sqrt{2}$.

It remains to consider how a deterministic algorithm can use the noisy prediction. Since the prediction has only two possible values, the algorithm's behavior is one of four deterministic strategies: follow the prediction, invert the prediction, always wait, or always acknowledge immediately. If the algorithm follows the prediction, then it makes the correct decision with probability $q$ in each phase. Therefore, its expected cost in each phase is at least $\big(q + (1 - q)\sqrt{2}\big)\mathrm{OPT}_i$, where $\mathrm{OPT}_i$ is the offline optimum cost in that phase. Hence, the expected ratio is $q + (1 - q)\sqrt{2} = \sqrt{2} - q(\sqrt{2} - 1) = \frac{2+\sqrt{2}-q}{\sqrt{2}+1}$. If the algorithm inverts the prediction, then it is correct with probability $1 - q$ and incorrect with probability $q$. Its expected ratio is therefore at least $(1 - q) + q\sqrt{2}$, which is larger than $q + (1 - q)\sqrt{2}$ because $q > 1/2$.

It remains to consider the two strategies that ignore the prediction. If the algorithm always waits, then it is correct in type-0 phases and incorrect in type-1 phases. Its expected cost per phase is at least $p \cdot \mathrm{OPT}_0 + (1 - p) \cdot \sqrt{2}\mathrm{OPT}_1 = (\sqrt{2} - 1)\sqrt{2}\rho + (2 - \sqrt{2})\sqrt{2}\rho = \sqrt{2}\rho$. The expected offline cost per phase is $p \cdot \mathrm{OPT}_0 + (1 - p) \cdot \mathrm{OPT}_1 = (\sqrt{2} - 1)\sqrt{2}\rho + (2 - \sqrt{2})\rho$. Thus, the

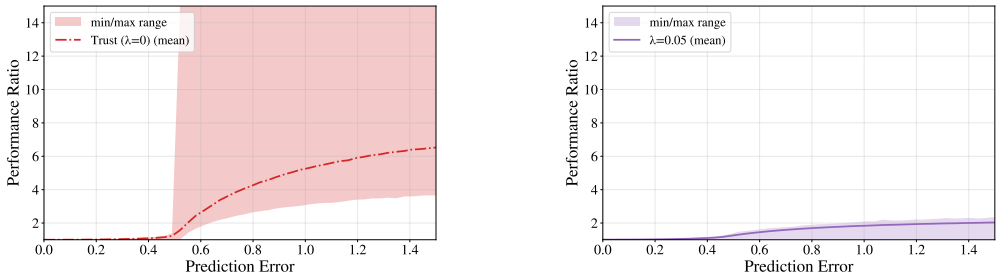
expected ratio is $\frac{\sqrt{2}\rho}{(\sqrt{2}-1)\sqrt{2}\rho+(2-\sqrt{2})\rho} = \frac{1+\sqrt{2}}{2}$. The same bound is obtained if the algorithm always acknowledges immediately: in that case, it is incorrect in type-0 phases and correct in type-1 phases, yielding expected ratio $\frac{(\sqrt{2}-1)\cdot 2\rho+(2-\sqrt{2})\rho}{(\sqrt{2}-1)\sqrt{2}\rho+(2-\sqrt{2})\rho} = \frac{1+\sqrt{2}}{2}$. Finally, since $q > 1/2$, we have $\frac{1+\sqrt{2}}{2} \geq q + (1-q)\sqrt{2} = \frac{2+\sqrt{2}-q}{\sqrt{2}+1}$.

Therefore, every deterministic algorithm has an expected cost of at least $\frac{2+\sqrt{2}-q}{\sqrt{2}+1}$ times the offline optimum under this phase-based distribution. □

## B Additional Experimental Details

**Variability across trials.** Figure 3 reports the performance ratio on real-world arrival sequences from the MAWI dataset as a function of prediction error $\eta$. Solid and dashed lines show the mean over 1000 independent trials, while shaded bands show the empirical min–max range. The results indicate that, for moderate values of $\lambda$, the Augmented$_{\text{Alarm}}(\lambda)$ family not only achieves low average performance ratios, but also exhibits relatively small variation across trials: the min–max bands remain close to the mean over a wide range of prediction errors. In contrast, baselines such as Trust, Fixed-time, and ACK-after-each show larger excursions under increasing prediction error. This supports the theoretical picture that moderate trust in predictions yields stable performance, while overly aggressive reliance on predictions can degrade sharply as $\eta$ increases.

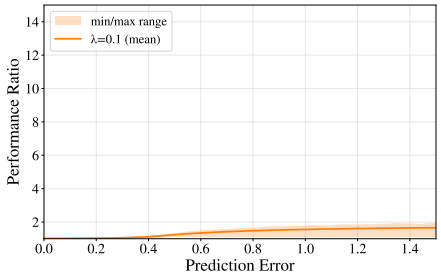
**Computational resources.** All experiments are lightweight CPU simulations and do not require GPU acceleration. They can be reproduced on a standard desktop or laptop machine, requiring less than 1GB of memory and completing in under an hour in total.



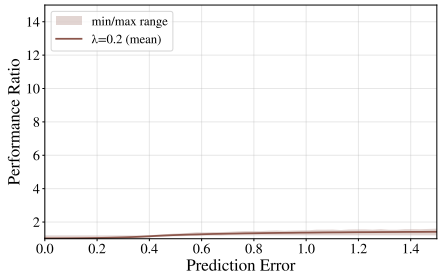

(a) Trust ($\lambda$ converges to 0)



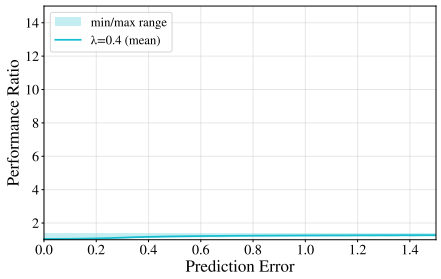

(b) $\lambda = 0.05$



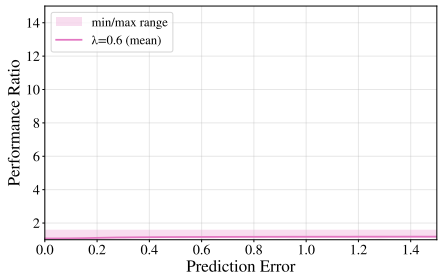

(c) $\lambda = 0.1$



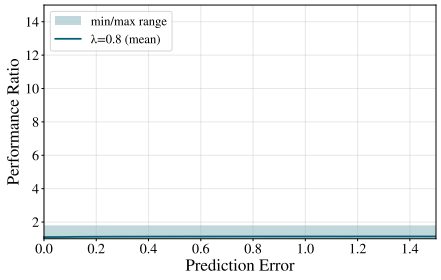

(d) $\lambda = 0.2$



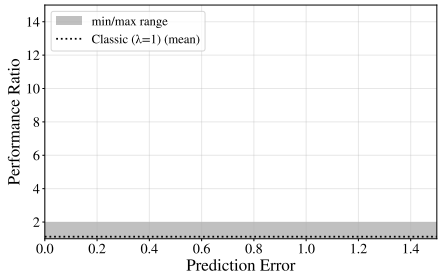

(e) $\lambda = 0.4$



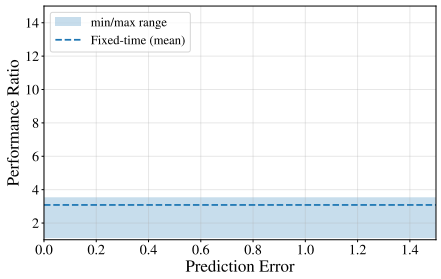

(f) $\lambda = 0.6$



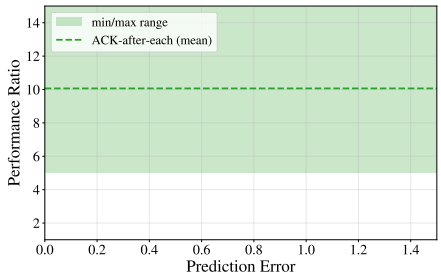

(g) $\lambda = 0.8$




(h) Classic ($\lambda = 1$)




(i) Fixed-time




(j) Ack-after-each

Figure 3: Performance ratios of the evaluated algorithms across varying levels of prediction error. Arrival patterns are derived from the MAWI dataset (min-max ranges are highlighted).