AFEC: An Adaptive Forward Error-Correction Protocol and Its Analysis

Kihong Park
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
park@cs.purdue.edu

CSD-TR 97-038
May 12, 1997

Abstract

This paper presents an adaptive protocol for packet-level forward error-correction in dynamic networks. The objective is to facilitate best-effort real-time applications whose timing constraints rule out the use of retransmission-based ARQ schemes. The degree of redundancy is adjusted as a function of network state, decreasing when the network is well-behaved and increasing when it is not. The control problem is nontrivial due to the fact that increased redundancy, beyond a certain level, backfires resulting in self-induced congestion which impedes the timely recovery of information at the receiver.

In the first part of the paper, we present a comprehensive analysis of the control problem associated with dynamic forward error-correction, concentrating on a particular protocol called Adaptive Forward Error-Correction (AFEC). We show that instabilities can arise from two distinct sources—desired operating point location and network delay—and we give solutions to handle them. The first causal factor is intimately tied to optimality, making its achievement potentially perilous in the context of QoS-greedy applications.

The second part of the paper presents simulation results that confirm the qualitative dynamics predicted by the analysis. We quantitatively estimate the redundancy-recovery rate function which relates redundancy to the quality of service rendered at the receiver. We show under what conditions the curve's shape is unimodal and to what degree. We compare the performance of AFEC against a static FEC protocol in which the redundancy factor is fixed. We show that AFEC exhibits superior performance when the network is subject to structural changes that persist for nonnegligible durations. Under short-range dependent traffic conditions, AFEC is able to closely match the performance of optimum static FEC but not exceed it.

1 Introduction

Forward error-correction (FEC) is a well-studied reliable[1] communication technique which has been successfully used, primarily at the bit-level, in a number of application domains from space communication to reliable data storage on compact disks [1, 15, 16]. The idea of FEC is simple: given a communication channel with known error probabilities, use error-correcting codes to inject redundant information into the transmission such that a desired recovery rate—i.e., probability of successful decoding at the receiver of code blocks—is achieved.

In the context of supporting multi-media traffic with real-time constraints over wide-area high-speed networks, packet-level FEC has generated renewed interest due to ARQ's inherent inadequacy at handling timing constraints when subject to large end-to-end network latencies [2, 4, 5, 10, 17, 20, 21, 26, 28]. Much of the previous research has concentrated on the issue of implementing efficient error-correcting codes at the software level from Reed-Solomon codes [18, 20] to other variants of polynomial codes over Galois Fields [17, 26], to ATM and TCP/IP over ATM implementation issues [2, 10]. Due to the low bit error rates associated with modern communication media, the assumption is made that decoding is mainly impeded by packet loss—i.e., *erasures*—caused by network congestion and the resultant buffer overflow. For the real-time applications we aim to support, delayed packet arrivals that violate the receiver's timing constraints are equally useless as if they had been dropped by the network.

The effectiveness of FEC is determined by two fundamental factors, *traffic burstiness* and *packet loss dependency*. In the former, given a fixed level of redundancy or overcode, the higher the burstiness, the higher the probability that information recovery will fail due to too many missing packets within a code block. This corresponds to an increase in channel error rate in the bit-level FEC model. "Burstiness" can be further characterized by taking the packet train [14] viewpoint whereby traffic sources are characterized by the interarrival time between successive *bursts*—a contiguous interval of densely spaced packets—the burst duration, and the burst size.

The effect of burstiness on FEC performance using this fine-grained characterization has been studied in a seminal paper [4] in the context of ATM networks. In [4], a *system* approach is taken in the sense that a group of FEC- and non-FEC sources and their interaction is investigated, and the aggregate performance is evaluated as a function of the specific group make-up and their individual traffic characteristics. A zero-sum law is shown to be at play whereby the least effective group

[1] In general, given stochastic channels or communication media, reliable transmission is achieved with "high probability" which stands in contrast to ARQ schemes which explicitly retransmit packets/bits until all information is received at the destination.

configuration is one where all sources use FEC resulting in a small or even negative aggregate gain.

This paper studies the second of the two factors that influence FEC effectiveness—*packet loss dependency*. The more correlated the packet loss at a bottleneck link, the more packets in a contiguous packet train will be dropped, thus causing a dimished recovery rate at the receiver. It is well-known that bit-level FEC is ineffective for dependent-error channels [11] whose implications also hold for packet-level FEC. In packet-level FEC, queueing behavior at bottleneck buffers is an intrinsic component of the system capable of producing complex dependencies among nearby and not-so-nearby packets in a traffic stream. Packet loss constitutes just one aspect of this more general dependency-generating facility. In [6, 8, 7], the impact of queueing-induced correlation on packet loss and delay in block-segmented traffic streams is analyzed. It is shown that vis-à-vis results stemming from an independent channel assumption [28], queueing-induced correlation produces a significant increase in packet loss rate and delay which has a negative impact on decoding success for FEC.

The modeling of bursts using packet trains is certainly a form of incorporating dependencies; however, in this paper we are interested in an even more specific form of dependency, namely, that induced by the redundancy itself when it is allowed to vary. To minimize coding overhead, it is reasonable to adjust the degree of redundancy as a function of network state, decreasing the overcode when the network is well-behaved and increasing it when the network is congested. The control problem is nontrivial due to the fact that increased redundancy, beyond a certain level, can backfire resulting in a form of self-induced congestion which impedes the timely recovery of information at the receiver. That is, an increase in redundancy causes the receiver's recovery rate to decrease.

The idea of adaptively employing FEC is not new. In [17], an "adaptive" scheme is proposed which assuming *known*, *independent* packet loss probability computes the overcode size needed to achieve a desired recovery rate. It constitutes calculating the probability that no more than h packets out of a total of $n > h$ get lost—a simple binomial sum due to independence—and is applicable to bit-level FEC scenario under independent-error channel assumption. However, it does not address dependency issues which are the heart of the packet-level FEC problem at hand. In [3], a fault-tolerant information recovery scheme supporting real-time constraints is proposed using a variant of Rabin's Information Dispersal Algorithm (IDA) [25]. Assuming *known* delay distributions to a set of information dispersal sites, schedules can be devised as a function of the latency distributions such that recovery of a document through the timely arrival of a sufficient number of code blocks is achieved with high probability. The impact of redundancy on network

network state and its implications to performance are not considered.

This paper's contribution is twofold. First, we formulate a model of the adaptive forward error-correction problem and present a comprehensive analysis of the associated optimal control problem. We propose a specific control algorithm called *Adaptive Forward Error-Correction* (AFEC) protocol and analyze its behavior with respect to optimality and stability. Our optimality criterion is based on *individual* optimality assuming the network characteristics, albeit stochastic, are fixed. The goal of the AFEC-controlled application is to achieve a desired recovery rate—its quality of service (QoS) requirement—when subject to time-varying network conditions and the influence of its own actions on network state. We show that instabilities can arise from two distinct sources—desired operating point location and network delay—and we characterize the conditions under which they can arise and give solutions to handle them.

Second, we present a simulation study of adaptive forward error-correction under various network conditions. We observe that the qualitative dynamics predicted by the analysis are confirmed in the simulation results. We quantitatively estimate the redundancy-recovery rate function which relates redundancy to the quality of service rendered at the receiver. We show under what conditions the curve's shape is *unimodal*—i.e., bell shaped—and to what degree. We compare the performance of AFEC against a static or nonadaptive FEC protocol in which the redundancy factor is fixed. We show that AFEC exhibits superior performance when the network is subject to structural changes that persist for nonnegligible durations, for example, as induced by sudden entries and departures of application pools sharing the same network resources. Under short-range dependent traffic conditions, however, AFEC is able to closely match the performance of *optimum* static FEC but not exceed it. Here, optimality is with respect to recovery rate achieved under different fixed redundancy values. This phenomenon is related to the fact that a feedback control is only as responsive as the time lag associated with the feedback loop allows it (e.g., round trip time). Burst activities generated by short-range traffic sources, by definition, are isolated in time with respect to correlation, and either anticipating or exploiting the occurrence of such bursts is an intrinsically difficult problem for any feedback control algorithm.

The remainder of the paper is organized as follows. In Section 2 we give the network model including the basic set-up of the adaptive forward error-correction problem. This is followed by a description of the AFEC protocol and the definition of the optimal control problem it is supposed to solve. Section 4 gives an analysis of stability including the description of an augmented AFEC protocol needed for a certain stability problem. Section 5 presents simulation results including an estimation of the redundancy-recovery relation, AFEC vs. static FEC comparisons, AFEC

dynamics, and its performance. We conclude with a discussion of current and future work.

2 Network Model

The goal of adaptive forward error-correction is to adjust the level of redundancy as a function of network state, decreasing it when the network is well-behaved and increasing it when the network is congested. Thus the overhead incurred by FEC is kept to a minimum. What makes the problem nontrivial is the fact that blindly increasing the level of redundancy will cause a congested system to become even worse, thus ultimately bringing forth a net decrease in the recovery rate. Furthermore, this positive feedback loop has the potential—depending on the control algorithm—to further increase the level of redundancy which only worsens the recovery rate. This is fundamentally different from bit-level FEC where the channel error rate is decoupled from redundancy although the error process itself is allowed to be correlated.

For simplicity of discussion and to focus on the main ideas, we will assume an ATM framework where packets are of fixed size—i.e., *cells*—and k data packets are encoded as $n = k + h$ cells where $h \geq 0$ is the degree of redundancy. We will assume that the receipt of any k cells out of the n total cells suffices to recover the original k data cells. Encoding/decoding functions with this property clearly exist (e.g., Reed-Solomon [18], IDA [25]).

Figure 1 depicts the shared output buffer at a bottleneck link where $\lambda^A(t)$ is the FEC-controlled application process, $\lambda^C(t)$ is the cross traffic, and $\mu(t)$ is the service process. The application in question is real-time constrained and the sender transmits

$$n(t_1), n(t_2), \ldots, n(t_i), \ldots \qquad (t_i < t_j \quad \text{if } i < j)$$

blocks of cells at times t_i where $n(t_i) = k(t_i) + h(t_i)$, $i = 1, 2, \ldots$. That is, $k(t_i)$ is the traffic requirement at t_i as dictated by the application and $h(t_i)$ the corresponding redundancy factor.

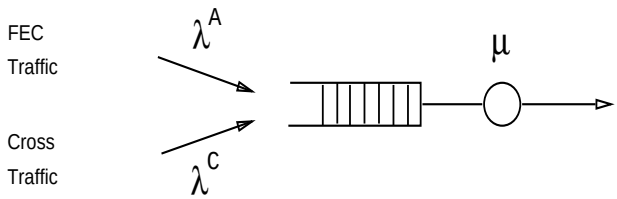

Figure 1: Shared queue at bottleneck link with FEC-controlled application traffic λ^A and cross traffic λ^C.

We will model quality of service (QoS) requirements using *hard* real-time constraints whereby we assume the existence of a monotonically increasing sequence $(t'_i)_{i\in\mathbb{Z}_+}$ of time deadlines at the receiver such that all $k(t_i)$ data cells belonging to the i'th block must be recovered before time t'_i. That is, late arrivals belonging to block i carry zero utility.

QoS is measured at the receiving end using a *recovery rate* process $\gamma(t'_i)$. It is defined to be the number of cells belonging to block i received before time t'_i. We will say there is a *hit* at time t'_i if $\gamma(t'_i) \geq k(t_i)$; i.e., decoding was timely and successful. We will extend γ's domain of definition to $\mathbb{R}_+$ by saying that for t such that $t'_i < t < t'_{i+1}$, for some $i \in \mathbb{Z}_+$, $\gamma(t)$ denotes the number of cells received belonging to block $i+1$ before time t. Thus for all $i \in \mathbb{Z}_+$, $\gamma(t)$ is a nondecreasing function on $(t'_i, t'_{i+1}]$.

The aforementioned model is general enough to represent a wide variety of real-time applications including those with periodic constraints as well as interactive applications with aperiodic constraints.

3 AFEC Protocol

Going back to our earlier remark, increasing $h(t)$ blindly will adversely affect $\gamma(t+\tau)$, for some $\tau > 0$. That is, letting $\mathcal{G}$, $\gamma(t+\tau) = \mathcal{G}(h(t))$, denote the functional relationship between $h(t)$ and $\gamma(t+\tau)$, there is a value $h^* > 0$ such that

$$\frac{d\mathcal{G}}{dh} = 0 \quad \text{at } h = h^*, \qquad \frac{d^2\mathcal{G}}{dh^2} < 0 \quad \text{for } h \neq h^*.$$

That is, $\mathcal{G}(h) \geq 0$ is unimodal with peak at $h = h^*$. Thus if the goal is to achieve maximum recovery rate, then we arrive at an optimal control problem where the optimal operating point is defined as (h^*, γ^*), $\gamma^* = \mathcal{G}(h^*)$. Figure 2 depicts the unimodal redundancy-recovery rate relation $\gamma = \mathcal{G}(h)$.

If $\gamma^* < k$, then there is a structural problem and no amount of redundancy, small or large, can yield a positive hit rate. This may be due to excessive cross traffic λ^C, long end-to-end delays, and a host of other factors.

Let us consider the case when $\gamma^* \geq k$. Assuming $\gamma(t)$ is fed back to the sender from the receiver, we can formulate the following control law

$$\frac{dh(t)}{dt} = \epsilon(\gamma_* - \gamma(t-\tau)) \tag{1}$$

where γ_*, $k \leq \gamma_* \leq \gamma^*$, is the *target* recovery rate, $\epsilon > 0$ is an adjustment parameter, and $\tau \geq 0$ is a delay term introduced by feedback and network latency. The control algorithm as embodied by

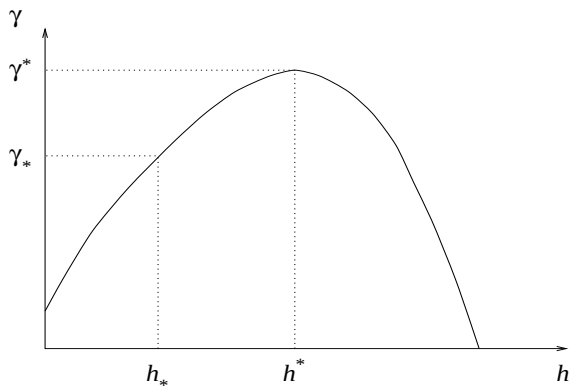

Figure 2: Unimodal redundancy-recovery rate function $\gamma = \mathcal{G}(h)$ with optimal recovery rate γ^* and target recovery rate γ_*.

(1) just says that if the measured recovery rate $\gamma(t-\tau)$ is smaller than the target recovery rate γ_*, then the redundancy factor h should be increased, and vice versa.

The larger the gap $\theta = \gamma_* - k > 0$, the more resiliently—in the presence of perturbations and sudden onset of increased burstiness—adaptive redundancy control can be affected. θ can be viewed as a yardstick of *conservativeness*; at the expense of increased overhead, higher resiliency can be achieved with respect to sudden changes in packet loss and delay.

When k is variable—as determined by a stochastic application arrival process $(k(t_i))_{i\in\mathbb{Z}_+}$—we yield a companion control law for (1) in the form of

$$\frac{dh(t)}{dt} = \epsilon'(\nu_* - \nu(t-\tau)) \tag{2}$$

where $\nu_* \geq 1$, $\epsilon' > 0$ is an adjustment parameter, and $\nu(t-\tau) = \gamma(t-\tau)/k(t_i)$ for i such that $t'_{i-1} < t - \tau \leq t'_i$. That is, i is the block number with respect to which the recovery rate $\gamma(t-\tau)$ was computed at the receiver. In practice, feedback is sent only at times $t = t'_1, t'_2, \ldots$, hence we have $t - \tau = t'_i$.

Setting $\nu_* = 1$ is tantamount to trying to achieve "perfect" redundancy in the sense that $dh/dt = 0$ iff $\gamma(t-\tau) = k(t_i)$; i.e., just the right number of cells have made it through to achieve recovery. Under stochastic network conditions such as when the variance of the cross traffic process $\lambda^C(t)$ is nonnegligible, ν_* needs to be increased to achieve higher hit rates. The gap $\nu_* - 1$, analogously to $\theta = \gamma_* - k$, is a measure of conservativeness.

The implementation of AFEC in the form of (2) is straightforward. On the receiver side, the protocol needs to maintain a counter for $\gamma(t)$, incrementing it whenever a new cell belonging to the expected block arrives. At time t'_i, $\gamma(t'_i)$ and the block sequence identifier i are sent back to

the sender and γ is reset to 0. On the sender side, given k and h at time $t = t_i$, the $k(t_i) + h(t_i)$ cells are transmitted. When a feedback control packet with $(\gamma(t_i'), i)$ is received, $h(t_{i+1})$ is updated based on $\gamma(t_i')$ and $k(t_i)$ using the discrete counterpart of (2). The value $k(t_i)$ is remembered by the sender in conjunction with index i.

Since (1) and (2) are related to each other via $\epsilon = \epsilon'/k$, when analyzing the behavior of the system under average application arrival rate conditions, it suffices to study the dynamics of (1).

4 Analysis of Stability

4.1 Operating Point Instability

Note that the redundancy-recovery relation, $\gamma(t) = \mathcal{G}(h(t - \kappa))$, is also tied to a delay term $\kappa \geq 0$. Hence (1) may be rewritten as

$$\frac{dh(t)}{dt} = \epsilon(\gamma_* - \mathcal{G}(h(t - \tau'))) \tag{3}$$

where $\tau' = \tau + \kappa$ ignoring the processing delay at the receiver. Equation (3) is a nonlinear autonomous delay differential equation[2] [9] and the study of its behavior is the subject matter at hand.

Two forms of instability may arise, one from the value of the desired operating point (h_*, γ_*), and another stemming from the delay term τ' in equation (3). First, assume $\tau' = 0$. Complications arising out of $\tau' > 0$—feedback always incurs a non-zero network delay—are orthogonal to the first instability consideration and will be dealt with separately.

To analyze $dh(t)/dt = \epsilon(\gamma_* - \mathcal{G}(h(t)))$, we will first linearize it and analyze the stability of the linearized system around a rest point. Assuming sufficient smoothness of $\mathcal{G}$ (mainly $\mathcal{G} \in C^1$), we can use the Stable Manifold Theorem [24] to relate the stability of the linearized system to that of the original nonlinear system in a local neighborhood of a rest point. That is, the (in)stability of the nonlinear system is implied by the (in)stability of the linearized system.

The Taylor expansion of the right-hand-side (RHS) of (3) around $h = h_*$ results in

$$\frac{dh}{dt} \;=\; \text{const} - \epsilon\mathcal{G}'(h_*)h \;+\; \text{higher order terms} \tag{4}$$

where $\mathcal{G}'(h_*) \equiv (d\mathcal{G}/dh)|_{h=h_*}$. The stability of the linearized system is dependent on the sign of $-\epsilon\mathcal{G}'(h_*)$, in particular, $-\epsilon\mathcal{G}'(h_*) < 0$. Since $\epsilon > 0$, we arrive at the stability condition of (3) (with

[2] Delay differential equations are also known as retarded functional differential equations.

$\tau' = 0$) which depends on the gradient $\mathcal{G}'(h_*)$: (3) is asymptotically stable at (h_*, γ_*) if

$$\mathcal{G}'(h_*) > 0. \tag{5}$$

By the unimodality of $\mathcal{G}$, this immediately gives us the corollary that (3) is asymptotically stable at $(h_*, \gamma_*) \in H_L$ and unstable at $(h_*, \gamma_*) \in H_R$ where

$$\begin{aligned} H_L &= \{(h, \gamma) : h = \mathcal{G}(h),\, h < h^*\}, \\ H_R &= \{(h, \gamma) : h = \mathcal{G}(h),\, h \geq h^*\}. \end{aligned}$$

In particular, the "optimal" operating point (h^*, γ^*), from the perspective of maximum recovery rate, is unstable.

Figure 3 (left) shows the phase portrait in the local neighborhood of (h^*, γ^*); we observe that for $h > h^*$ there is divergence. Figure 3 (right) shows the corresponding phase portrait for $(h_*, \gamma_*) \in H_L$. Clearly, the flow to (h_*, γ_*) is convergent. In the case of the rest of the unstable set, $(h_*, \gamma_*) \in H_R \setminus \{(h^*, \gamma^*)\}$, the flow is again divergent for $h > h_*$, analogous to Figure 3 (left); it is omitted here.

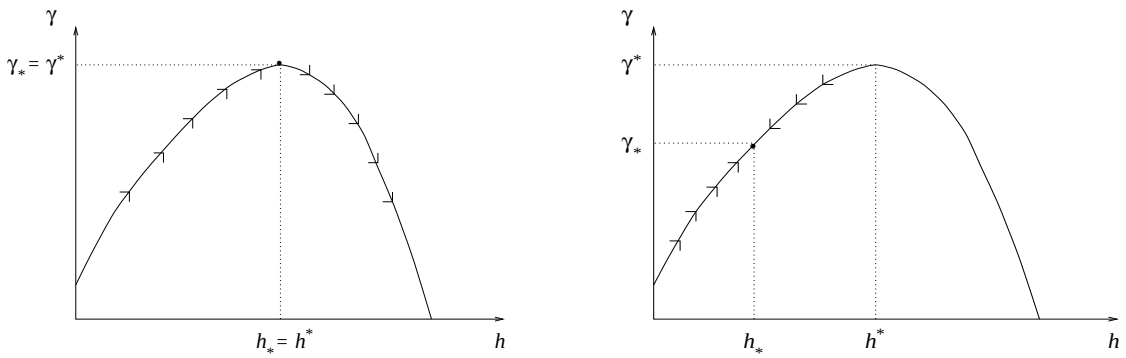

Figure 3: Left: Phase portrait around the unstable fixed point $\gamma = \gamma^*$. Right: Phase portrait around the stable equilibrium $\gamma = \gamma_* < \gamma^*$.

4.2 Augmented AFEC Protocol

The previous section has shown that instability ensues when "too much" redundancy is applied at the sender which is tantamount to shooting oneself in the foot. In fact, the reasons underlying the instability of (h^*, γ^*) are conceptually similar to the ones underlying a *congestion susceptible* system—a two-dimensional system—which has been studied in the context of congestion control in [23]. Whereas in the latter the natural goal is to achieve maximum utilization, in the FEC context,

(h^*, γ^*) carries with it a large redundancy overhead and the true optimal operating point may be at some value $h_* < h^*$ which meets the desired hit rate requirement at minimum redundancy overhead.

To achieve a form of stability at (h^*, γ^*) which may be needed when the difference $\gamma^* - k$ is not large or the variance of the system high, one can augment the control law given in (1) via a *directional check* given by the sign of $d\gamma/dh$. If $d\gamma/dh > 0$, then the system is in the stable region $(h < h^*)$ and (1) is applied as usual. If, on the other hand, $d\gamma/dh \leq 0$, then the system finds itself in the unstable region $(h \geq h^*)$ and a *backoff mechanism*—$dh/dt < 0$—is instituted until $d\gamma/dh > 0$ at which time we find ourselves again in the stable regime.

The Augmented AFEC algorithm is given by

$$\frac{dh(t)}{dt} \;=\; \epsilon(\gamma_* - \gamma(t-\tau))\; sgn\left(\frac{d\gamma(t-\tau)}{dh}\right) \tag{6}$$

where $sgn(x) = 1$ if $x \geq 0$, and $sgn(x) = -1$, otherwise. Hence (6) follows (1), as it should, when $h \leq h^*$, and it performs a backoff when $h > h^*$. Another variant uses an exponential backoff scheme—$dh/dt = -ah$, $a > 0$—which leads to an exponential decay of h of the form $O(e^{-at})$. Implementing directional detection—i.e., estimating derivatives, in general—can be tricky; we use information maintained at the sender in the form of a local history $(i, h(t_i))$ in conjunction with information contained in the feedback $(i, \gamma(t_i'))$ for the estimation of $sgn(d\gamma(t-\tau)/dh)$. There is again an observation delay incurred whose general effect is studied in Section 4.3.

We remark that a control law such as

$$\frac{dh}{dt} = \epsilon(h_* - h) \tag{7}$$

would render *all* $(h, \mathcal{G}(h))$ asymptotically stable hence obviating the need for the Augmented AFEC protocol. However, whereas (1), (2) are feasibly implementable control algorithms since both k and γ are known and measurable quantities, respectively, (7) depends on a more elusive quantity h_* whose estimation is contingent upon quantitatively knowing the redundancy-recovery rate function $\mathcal{G}$, a questionable assumption in the context of dynamic networks and on-line control.

4.3 Delay Instability

We now return to the issue of incorporating the effect of network delay as captured by $\tau' > 0$ in equation (3) with respect to stability. We will show that the presence of network delay, in general, can lead to oscillatory behavior even for $(h_*, \gamma_*) \in H_L$ for which we have shown asymptotic stability

when $\tau' = 0$ (Section 4.1). If τ' is sufficiently large, then the oscillation can become unbounded leading to a different causal aspect of instability.

We use a modified form of the Stable Manifold Theorem, adopted to the case of nonlinear autonomous delay differential equations [9], to relate the stability of the nonlinear system to the stability property of the linearized system around a rest point. We assume sufficient smoothness and boundedness on $\mathcal{G}$. The stability of the linearized system corresponding to (3) in the neighborhood of $(h_*, \gamma_*) \in H_L$ is completely determined by the stability of the reduced linear system

$$\frac{dh(t)}{dt} = -\epsilon\, \mathcal{G}'(h_*) h(t - \tau'). \tag{8}$$

A linear delay differential equation of the form $dx(t)/dt = -ax(t - \tau)$, $a > 0$, has a sinusoidal solution which is determined by the complex roots of the characteristic equation $\xi + ae^{\xi\tau} = 0$ [13]. The system is asymptotically stable if for all roots of the characteristic equation, the real parts lie in the negative half plane. The latter, in turn, is related to the relative magnitude of a and τ which is shown next.

By a change of variables, we further reduce (8) to a normal form where $\tau' = 1$. This is useful because the stability condition of $dx(t)/dt = -ax(t - 1)$ can be shown to be $0 < a < \pi/2$ [13]. Let $s = t/\tau'$. Then,

$$\frac{dh(\tau' s)}{ds} = -\tau'\epsilon\, \mathcal{G}'(h_*) h(\tau'(s - 1)),$$

from which we get

$$\frac{d\tilde{h}(s)}{ds} = -\tau'\epsilon\, \mathcal{G}'(h_*) \tilde{h}(s - 1) \tag{9}$$

where we have redefined $\tilde{h}(s) \equiv h(\tau' s)$. Thus we arrive at the asymptotic stability condition for the original nonlinear system (3) via its relation to (9),

$$0 \;<\; \tau'\epsilon\, \mathcal{G}'(h_*) \;<\; \frac{\pi}{2}. \tag{10}$$

Hence, the larger the network delay τ', the smaller the adjustment factor ϵ needs to be to preserve stability. The gradient $\mathcal{G}'(h_*)$ stands at a similar trade-off relationship with ϵ with respect to stability.

5 Simulation Results

5.1 Set-Up

The simulation set-up is based on the network configuration shown in Figure 4. A is the application node which generates the FEC-controlled source traffic $\lambda^A(t)$, C is the cross traffic node with arrival

process $\lambda^C(t)$, and G is a gateway implementing an output-buffered switch with FIFO service, as shown in the logical diagram, Figure 1. D is the destination node at which the quality of service with respect to the hit rate ν is measured. The links are unidirectional (digraph) with each edge $X \leftrightarrow Y$ representing two separate links (X, Y) and (Y, X). Quantities of interest include the traffic flow on the bottleneck link (G, D), the queueing behavior at its output buffer, and the arrival times of the $\lambda^A(t)$ process packets at D which determines the QoS received by the application.

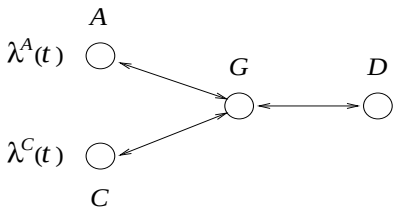

Figure 4: Bottleneck network set-up with FEC-controlled data source A, cross traffic source C, gateway G with outputbuffered switch, and destination node D; λ^A, λ^C represent the application traffic process and cross traffic process, respectively.

The FEC-controlled application—a pair of processes, i.e., the sender at A and the receiver at D—constitutes a full-duplex end-to-end connection from A to D and vice versa. The application is modeled as a periodic hard real-time system whereby $k(t_1), k(t_2), \ldots, k(t_i), \ldots$, data packets, $t_{i+1} = t_i + T$ for some $T > 0$, are generated at A, and the $k(t_i)$ data packets must be recovered at D before time t'_i where $t'_{i+1} = t'_i + T$. An example would be a video-conferencing application, compressed[3] or uncompressed, whereby to meet the f frames per second requirement, $k(t_i)$ data packets must be recovered successfully during the interval $(t'_{i-1}, t'_i]$. Moreover, $t'_i \geq t_i + \ell$ where $\ell > 0$ is the minimum network latency from A to D.

With respect to scheduling, sender A has two degrees of freedom. One, when communicating the information contained in the i'th block of $k(t_i)$ packets, A may choose to use an error-correcting code yielding an amplified block consisting of $n(t_i) = k(t_i) + h(t_i)$ packets where $h(t_i) \geq 0$ represents the redundancy level or overcode. The only requirement is that the decoding function must be able to reconstruct the original $k(t_i)$ data packets from *any* $k(t_i)$ of the $n(t_i)$ code packets transmitted (cf. Section 2). If $X_i = (x_{i1}, x_{i2}, \ldots, x_{in_i})$, $n_i = n(t_i)$, denotes the sequence of packets generated at A belonging to block i, and $Y_i = (y_{i1}, y_{i2}, \ldots, y_{im_i})$, $y_{ij} \in X_i$, denotes the packets received at D during $(t'_{i-1}, t'_i]$, we will say a *hit* has occured at time t'_i if $m_i \geq k(t_i)$.

[3]When using MPEG, a minor adjustment is needed to handle interframe dependencies, in particular, those induced by B frames.

Two, A may choose to schedule the $n(t_i)$ packets *in time*, i.e., over the interval $(t_{i-1}, t_i]$. A function $\pi : [t, t+T] \rightarrow \{0,1\}$ such that $\sum_{s\in[t,t+T]} \pi(s) = h$ will be called a *time-redundancy schedule* over $[t, t+T]$ with weight h. Our results are based on uniform π which, to some extent, achieve decorrelation to counteract the tendency of packet drops to occur in bursts. A third scheduling variable is to use *dispersity routing* [19]—a form of *space-redundancy scheduling*—to achieve decorrelation over space; the latter, however, requires assistance from the internetworking layer to access routing decisions. In this paper, we concentrate on pure end-to-end mechanisms.

5.2 Redundancy-Recovery Rate Relation

Figure 5 shows the measured redundancy-recovery rate relation $\gamma = \mathcal{G}(h)$ for the previous network configuration, expressed as the normalized quantity $\nu = \mathcal{G}(h)$ where $\nu = \gamma/k$ is the hit rate. The cross traffic $\lambda^C(t)$ was a Poisson process with rate $\lambda^C = 10$ cells per unit time, service rate $\mu = 12$ cells/unit time, code block size $k = 2$, application receive period $T = 4$, and link latency 1. The processing time at the nodes was assumed to be fixed (0 in this case). Each measurement was obtained from a simulation run of 3000 time units. Keeping the output buffer B at gateway G fixed, traces were collected when the redundancy factor h was held constant[4] at the start of the run. h was varied in the range $h = 0, 1, 2, \ldots, 15, 20, 25, 30$, and the collection of runs corresponding to three buffer capacities $B = 15, 55, 65$ are shown in Figure 5.

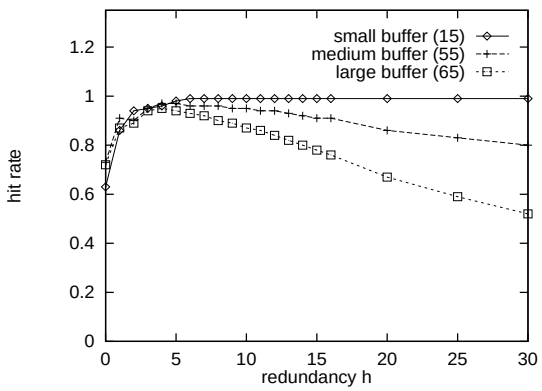

Figure 5: Observed redundancy-hit rate relation $\nu = \mathcal{G}(h)$ (note, $\nu = \gamma/k$) measured under three bottleneck buffer capacity (in cells) conditions: 15 cells (small), 55 cells (medium), and 65 cells (large).

For $B = 15$ ("small buffer size"), we observe the hit rate ν increasing as h is increased, saturating

[4]That is, a static or nonadapative FEC protocol, not AFEC, was used for the measurement process.

thereafter. That is, contrary to the unimodal shape assumed of $\mathcal{G}$—cf. Figure 2—the hit rate, nor the recovery rate γ, decreases for large h. This phenomenon stems from the fact that small buffer capacities imply a form of "forgetfulness" which induces an extreme case of many-to-one mapping from the space of traffic characteristics to the output buffer state. In particular, increasing the influx rate (or burstiness)—in this case h—beyond a certain point saturates mean queue length, and since the sender-side of the application transmits new blocks every T time units, the net recovery rate at the receiver-side saturates as well, yielding the flat hit rate curve observed in Figure 5. The implications of small buffer capacities and time horizons on network performance under long-range dependent traffic conditions have been investigated in [12, 27].

For $B = 55$ ("medium buffer size"), we see a decline in the hit rate as h is increased beyond a certain point, $h = 6$, continuing to decrease thereafter. This is even more pronounced for $B = 65$ ("large buffer size"[5]), where for $h = 30$, a hit rate is achieve that is even smaller than the hit rate achieved with no redundancy. For $B = 55, 65$, the mean dwell time in the queue has become significant enough vis-à-vis the application's receive period T so that further increases in h boost the probability that a packet belonging to block i is delayed in the queue by packets belonging to block $i - 1$. This, in turn, adversely affects the average timeliness of packets—even if the packets in question have not been dropped—causing a decline in the recovery rate.

The curvature of $\mathcal{G}$ further increases with B thus justifying the unimodal shape assumption qualitatively shown in Figure 2.

5.3 Static vs. Adaptive FEC

Figure 6 shows a typical scenario where the relative advantage of adaptive FEC vs. static or nonadaptive FEC can be discerned. Assume we are given an application that can tolerate a hit rate of $\nu = 93\%$. The network was configured with bottleneck buffer capacity $B = 55$ cells, Poisson cross traffic with rate 12, service rate 13, receive period $T = 4$, and application block size $k = 2$. Using a priori information about the network state including empirical measurements, assume the application was prevy to the information that a redundancy rate fixed at $h = 3$ would suffice to achieve the desired hit rate of 93%. For present purposes, the $h = 3$, $\nu = 93\%$ correspondence for this particular network configuration was obtained by extensive measurement, as was done in Figure 5, to estimate the shape of the redundancy-recovery rate function $\mathcal{G}$.

[5]Note that there is only a size 10 gap between the "medium" and "large" buffer sizes whereas there is a size 40 difference between the "medium" and "small" sizes. The sensitivity of the curvature of $\mathcal{G}$ on buffer capacity B increases with B.

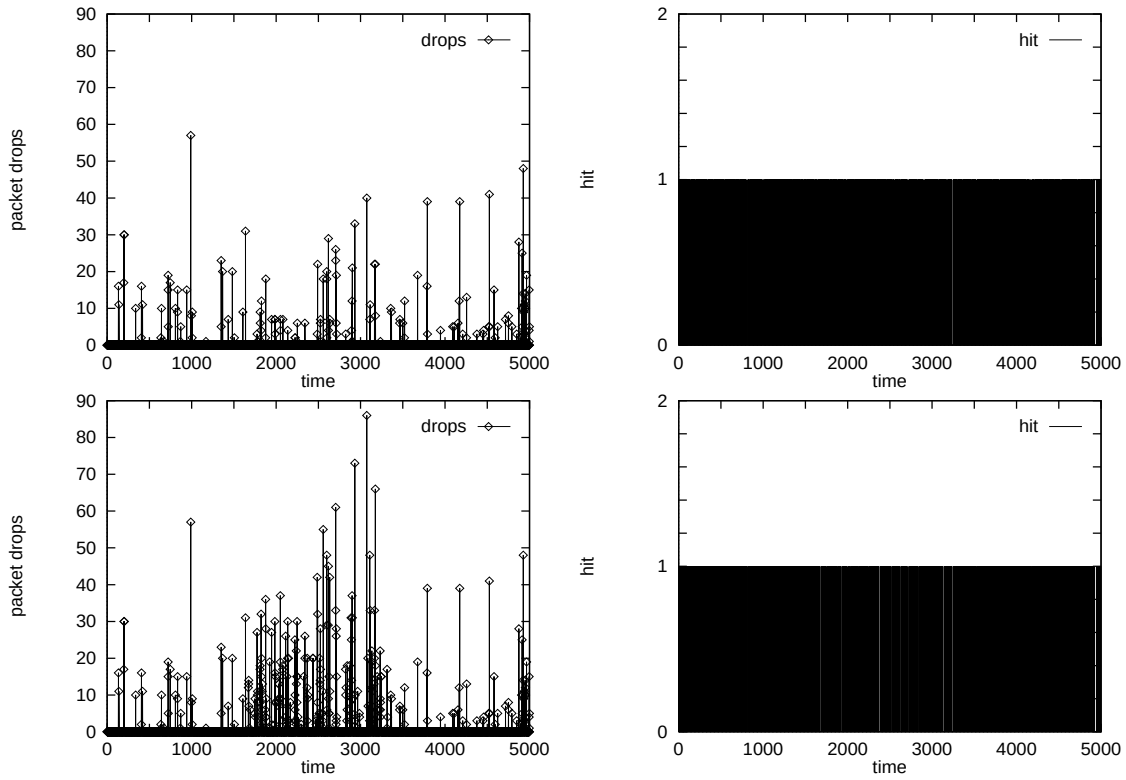

Figure 6: Static FEC run with constant redundancy $h = 3$. Top row: Packet loss trace (left) and hit occurrence trace at receiver (right) for 5000 time steps. Bottom row: Same as above but with the difference that the background traffic (λ^C) is increased at time $t = 1700$ and decreased to its original level at $t = 3400$. The *white* stripes in the right-column figures indicate instances at the receiver when its real-time constraints were violated.

The top row of Figure 6 shows the packet drop trace at the bottleneck buffer (left) and hit trace at the receiver (right) for a 5000 time unit trace. Note that the presence of a black unit impulse at $t = t'_i$ in the hit trace implies that at least $k(t_i)$ cells were received at the destination D before time t'_i. That is, decoding was timely and successful. Although there is a strong correlation between packet drops and occurrence of *misses*—i.e., recovery failure—at the receiver, an isolated burst of packet drops need not necessarily affect the recovery of the corresponding data block at the receiver since we employ a uniform time-redundancy schedule π to spread out the redundancy in time. We observe a few concentrated periods of packet drops here and there—one especially prominent near the end of the run—which lead to corresponding recovery failures at the receiver.

The bottom row of Figure 6 shows the same set-up except that the cross traffic $\lambda^C(t)$ was perturbed during the middle third of the run. In particular, at time $t = 1700$, λ^C was increased

from 12, its previous level, to 15 thus increasing both its mean and variance. $\lambda^C(t)$ was returned to its previous characteristics at time $t = 3400$. The packet drop profile shows a marked increase in the loss rate during the $[1700, 3400]$ period, and a commensurate negative impact on the recovery rate during the same period. The static FEC continues to maintain its redundancy rate at $h = 3$, and the resulting hit rate—88%—significantly violates the application's QoS requirement. Perturbations such as these are common and to be expected in dynamic, shared network environments, and it points toward a weakness of applying FEC statically.

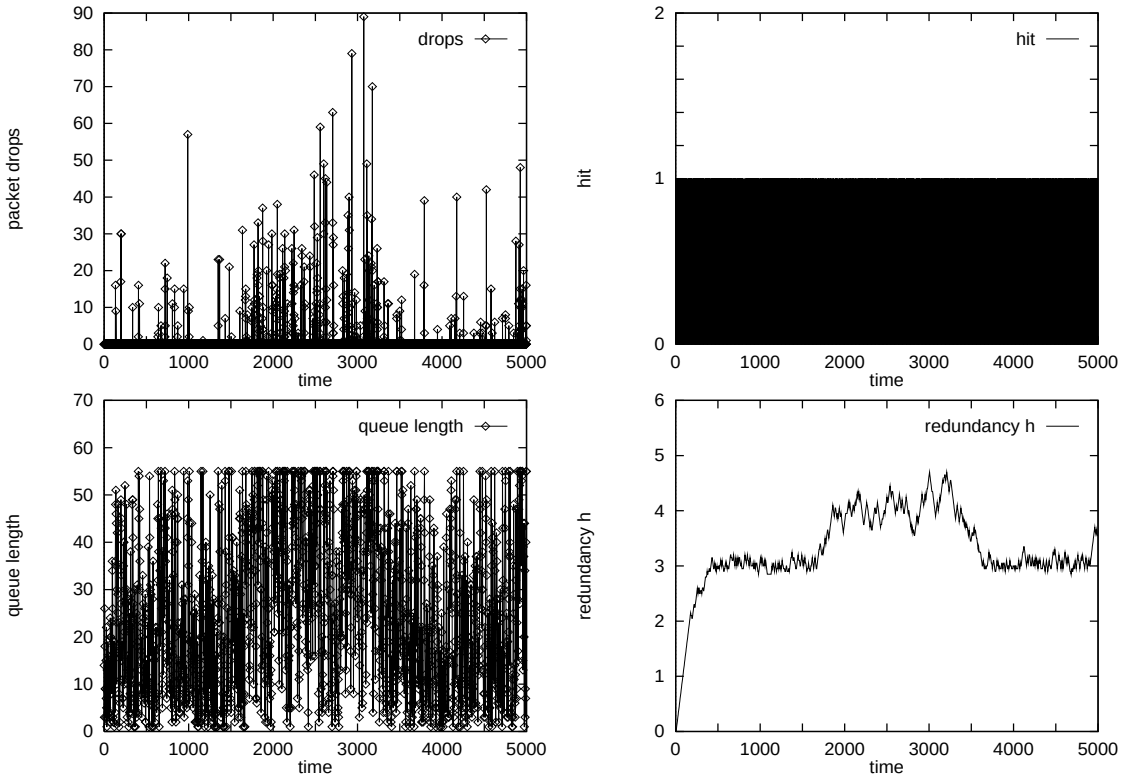

Figure 7: AFEC run. Top row: Packet loss trace (left) and hit occurrence trace at receiver (right) when the background traffic is increased at time $t = 1700$ and decreased to its original level at $t = 3400$. Bottom row: Buffer occupancy trace (left) and redundancy rate h trace (right) for the same run.

Figure 7 (top row) shows the corresponding packet drop and hit traces for the same network configuration when the traffic level was increased at $t = 1700$ and returned to its original level at $t = 3400$. The hit trace clearly shows a marked improvement over the corresponding trace of the static FEC run in Figure 6 (bottom row, right). The packet drop traces look similar due to the dominance of the cross traffic process $\lambda^C(t)$ in determining variability; the application's variability,

in the current set-up, is restricted to changes in h. The same cross traffic process—i.e., sample path—was used in both runs for comparability. Upon closer examination, we detect an elevation in the packet drop sequence during the interval $[1700, 3400]$ for the AFEC run vis-à-vis the static FEC run due to a higher value of h.

The reason for the improved performance can be gleamed from Figure 7 (bottom row, right) which shows redundancy rate h as a function of time[6]. During the "busy" interval $[1700, 3400]$, h is increased accordingly to compensate for the increased packet loss rate. Note that the increase in h, as noted above, further *increases* the overall packet loss rate during $[1700, 3400]$. However, the recovery rate at the receiver is in the range where it can still benefit from increased redundancy, thus resulting in an improved performance with respect to recovery rate vis-à-vis static FEC.

This is also reflected in the hit rate which was 94% in the case when there was no sudden increase in the cross traffic, and in the case when increased cross traffic perturbation is applied, AFEC is still able to achieve a hit rate of 93%. This is a marked improvement compared to static FEC where the hit rate had dropped from 93% to 88%. Thus, AFEC is able to achieve a measure of constancy in the presence of structural perturbations to the network state.

5.4 AFEC Dynamics

5.4.1 Operating Point Instability

In this section, we show the instability ensueing from applying control law (1) without the backoff mechanism incorporated in the augmented AFEC control equation (6). Figure 8 (left) shows the observed redundancy-hit rate function with network configuration settings where the bottleneck buffer capacity was set relatively high ($B = 85$). The maximum hit rate achievable by the static FEC protocol is 0.85. The Augmented AFEC protocol was able to achieve a hit rate of 0.84. The sensitivity of the curve near its peak and its early rise leave little room for adjusting h; i.e., $\gamma^* - k$ is very narrow.

Figure 8 (right) shows the h and buffer occupancy trace for a AFEC run without the backoff mechanism of (6). With the backoff mechanism turned off, we see a positive drift in h—still fluctuation up and down—until around time $t = 2750$ at which point the positive feedback loop kicks in with force, prompted by the magnitude of h becoming the dominating term with respect to total traffic influx. The queue quickly reaches saturation resulting in further decreases in the hit

[6]Note that h is maintained as a real variable even though redundancy can only be discrete, i.e., a nonnegative integer. This is to achieve better control resolution; the h value is rounded before determining the size of the actual overcode.

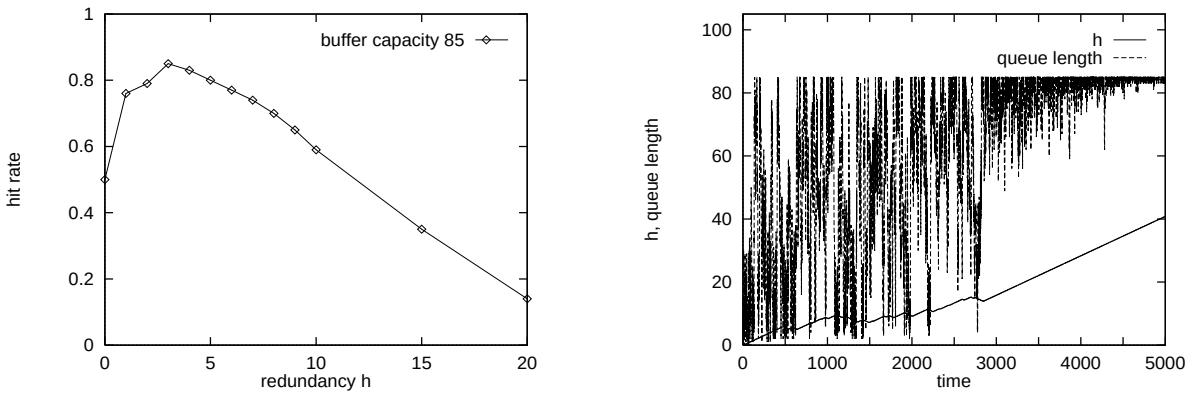

Figure 8: Left: Observed redundancy-hit rate function $\nu = \mathcal{G}(h)$ for large bottleneck buffer ($B = 85$) condition. Right: Redundancy h and buffer occupancy trace for the same system with a too large target recovery rate γ_*, $\gamma_* \approx \gamma^*$.

rate ν as predicted by the sensitive unimodal shape of the network's redundancy-hit rate function (Figure 8 (left)). This, in turn, provokes further increases in h by control law (1) sustaining the positive feedback loop since (1) is oblivious to the fact that it finds itself in the unstable region.

5.4.2 Delay Instability

Figure 9 shows the adverse effect of increased network delay in stabilizing system behavior as predicted by the analysis of Section 4.3. Figure 9 (left) shows the redundancy rate h trace for a typical network configuration where the feedback link (D, G) from the destination D to the router G was set at 10. The adjustment factor ϵ in control law (1) was fixed at $\epsilon = 0.15$.

Figure 9 (middle) shows the same system with the only difference that the link latency on the feedback link (D, G) was increased from 10 to 100. The marginal effect of increasing network delay is clearly discernible where the amplitude of the oscillation has increased several-fold, leading to a wildly fluctuating, ill-behaved system.

To further illustrate the stability condition captured in relation (10), i.e., $0 < \tau\epsilon < \text{const}$, we have kept everything the same as in the network configuration of the middle figure, except that the adjustment factor ϵ was decreased from 0.15 to 0.03. Its effect is shown in Figure 9 (right). As predicted by the stability condition (10), clearly the amplitude of the oscillation has damped down to a similar level as that of Figure 9 (left). One difference is the diminished responsiveness of the resulting system to changes in network state which is a price that must be paid to achieve stability.

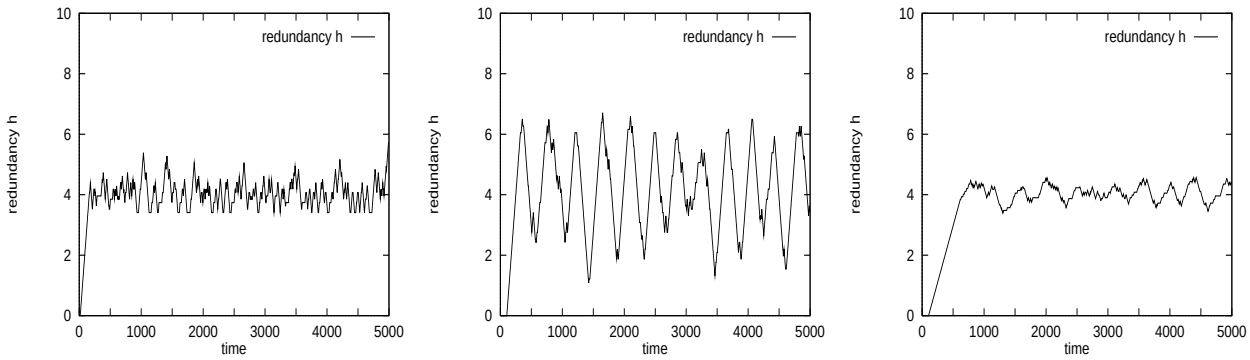

Figure 9: Left: Redundancy h trace for a network configuration with a small end-to-end network latency. Middle: Redundancy trace for the same network configuration except that the end-to-end link latency has been increased from 10 to 100. Right: Same network configuration as in the middle figure except that the adjustment factor ϵ was decreased.

5.5 AFEC Performance

Figure 10 (left) shows the hit rate ν of the AFEC protocol as a function of cross traffic intensity λ^C, $\lambda^C = 0, 1, \ldots, 10, 15, 20, 30, 40$ under three different bottleneck buffer capacity conditions, $B = 15, 55, 75$. When cross traffic intensity is low, the three performance curves coincide indicating an underutilized system where the packet loss rate is sufficiently small so that losses or delay fluctuations can be almost entirely compensated by a suitable infusion of redundancy. As the cross traffic intensity increases, however, adaptivity becomes less effective, especially for the large buffer case ($B = 75$) due to the pronounced unimodal shape of its redundancy-recovery rate function. For extremely high cross traffic intensity values, the performance graphs converge again. Even For best-effort real-time applications capable of tolerating nonnegligible miss rates, a hit rate below—say 80%—may be too small to be useful.

To discern how much of the performance loss is due to intrinsic limitations imposed by the increase in cross traffic intensity, we have measured the *best* performance achieved by the static FEC protocol where "best" refers to taking the maximum over the set of initial h values. Figure 10 (right) plots the optimum static FEC performance curve against AFEC's performance curve for $B = 55$. For very large cross traffic intensities we observe divergence setting in; however, for the postulated hit rates of interest (e.g., 80%), the two curves remain close together.

One may view the optimum static FEC performance graph as representing an *envelop curve* that delimits the boundaries of what is achievable. Although it is possible for AFEC to match the performance of optimum static FEC, it is difficult to exceed it under present cross traffic

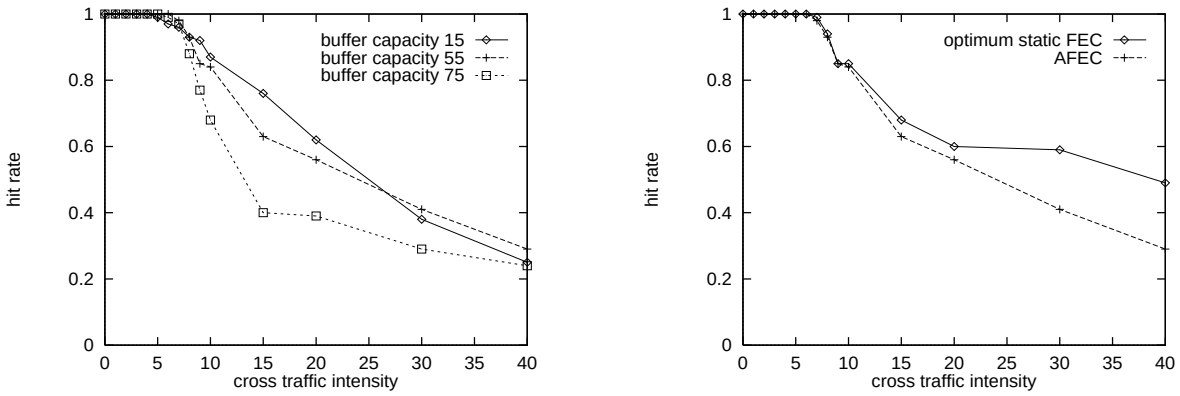

Figure 10: Left: Hit rate ν of AFEC as a function of traffic intensity $\lambda^C(t)$ under three buffer capacity conditions $B = 15, 55, 75$. Right: Same set-up where AFEC is compared against the *best* performance (over initial h) of static AFEC for $B = 55$.

conditions—Poisson arrivals with rate λ^C—due to the short-range dependent nature of the cross traffic source. That is, since AFEC is a feedback control protocol, it must observe the network state before it can adjust to it. Short-range dependent sources have the property that the occurrence of intense activity at time t bears little correlation to the activity level at time $t + T$ for $T > 0$ "large." Indeed, given control computations including encoding/decoding that must be carried out at the millisecond or submillisecond level, network latencies can be hundreds of milliseconds long, and the short-range correlation structure may be too brief relative to T (e.g., round-trip time) to be highly effective for adaptation purposes[7]. AFEC is most effective vis-à-vis static FEC when network changes are persistent as shown in Section 5.3.

Returning to Figure 10 (right), based on the previous discussion, for the hit rates of interest, AFEC may be achieving performance that is close to what is, in the best of circumstances, possible.

6 Conclusion

We have presented an analysis of the adaptive forward error-correction problem, from its formulation to a proposed control algorithm called Adaptive Forward Error-Correction (AFEC) protocol, to its analysis of optimality and stability. We have shown that two forms of instability can arise, and we have characterized their properties including methods for handling them.

We have shown simulation results which confirm the dynamics predicted by the analysis. We have estimated the quantitative shape of the redundancy-recovery rate relation for various network

[7] This is also related to the delay-bandwidth product which increases proportionately in broadband networks.

configurations and we have shown under what conditions the curve is unimodal. We have compared the performance of AFEC against a static or nonadaptive FEC control and we have shown that AFEC is superior to static FEC when the network is subject to structural changes of nonnegligible time duration. Under short-range dependent traffic conditions, AFEC is able to closely match the performance of the optimum static FEC where the maximum, with respect to performance, is taken over the set of initial redundancy values. However, AFEC does not exceed the optimum static FEC's performance.

This paper has pursued an end-to-end approach to achieving adaptive redundancy control for packet-level forward error-correction. The time-redundancy schedule π was set to uniform to combat the tendency of packet drops to occur in bursts. A further extension involves using space-redundancy scheduling in the form of dispersity routing [19] whereby decorrelation is achieved by choosing a set of disjoint or weakly correlated paths in the network. A drawback of this approach lies in the need to access routing functionalities thus deviating from the pure end-to-end solution taken thus far.

The bulk of the current work is directed at evaluating AFEC's performance under self-similar traffic conditions. We use a simulation set-up previously employed in the context of studying the causal and performance impact of long-range dependent traffic under various network conditions [22]. On the one hand, traffic self-similarity is bad news since bursts exist at multiple time scales and they do not dampen as traffic streams are multiplexed. On the other hand, long-range dependence suggests the presence of correlation structure which may be exploitable for adaptive forward error-correction purposes. A separate project involving the implementation of AFEC for a best-effort teleconferencing application is under way and we hope to report these results in the near future.

References

[1] E. Berlekamp, R. Peile, and S. Pope. The application of error control in communications. *IEEE Communications Magazine*, 25(4):44–57, 1987.

[2] A. Bestavros and G. Kim. TCP Boston: a fragmentation-tolerant TCP protocol for ATM networks. To appear in *Proc. IEEE INFOCOM '97*, 1997.

[3] Azer Bestavros. AIDA: A bandwidth allocation strategy for distributed time-critical real-time systems. In *Proc. IEEE IPPS Workshop on Parallel and Distributed Real-Time Systems*, 1993.

[4] Ernst Biersack. Performance evaluation of forward error correction in ATM networks. In *Proc. ACM SIGCOMM '92*, pages 248–257, 1992.

[5] J. C. Bolot and A. Vega-Garcia. The case for FEC-based error control for packet audio in the Internet. To appear in *ACM Multimedia Systems*, 1996.

[6] I. Cidon, A. Khamisy, and M. Sidi. Analysis of packet loss processes in high-speed networks. *IEEE Trans. Information Theory*, 39(1):98–108, 1993.

[7] I. Cidon, A. Khamisy, and M. Sidi. Dispersed messages in discrete-time queues: delay, jitter, and threshold crossing. In *Proc. IEEE INFOCOM '94*, pages 218–223, 1994.

[8] I. Cidon, A. Khamisy, and M. Sidi. On queueing delays of dispersed messages. *Queueing Systems, Theory and Applications*, 15:325–346, 1994.

[9] O. Diekmann, S. van Gils, S. Verduyn Lunel, and H. Walther. *Delay Equations: Functional-, Complex-, and Nonlinear Analysis*. Springer Verlag, 1995.

[10] S. Dravida and R. Damodaram. Error detection and correction options for data services in B-ISDN. *IEEE J. Select. Areas Commun.*, 9(9):1484–1495, 1991.

[11] F. C. Fujiwara, M. Kasahara, K. Yamashita, and T. Namekawa. Evaluations of error-control techniques in both independent and dependent-error channels. *IEEE trans. on commun.*, COM-26, 6:785–793, 1978.

[12] M. Grossglauser and J-C. Bolot. On the relevance of long-range dependence in network traffic. In *Proc. ACM SIGCOMM '96*, pages 15–24, 1996.

[13] Jack K. Hale and Sjoerd M. Verduyn Lunel. *Functional Differential Equations*. Springer-Verlag, 1993.

[14] R. Jain and S. Routhier. Packet trains—measurements and a new model for computer network traffic. *IEEE J. Select. Areas Commun.*, 4(6):986–995, 1986.

[15] S. Kaneda and E. Fujiwara. Single bit error correcting double bit error detecting code for memory systems. *IEEE Trans. on Computers*, C-31(6):596–602, 1982.

[16] K. Kawashima and H. Saito. Teletraffic issues in ATM networks. *Computer Networks and ISDN Systems*, 20:369–375, 1990.

[17] B. Lamparter, Otto Böhrer, W. Effelsberg, and V. Turau. Adaptable forward error correction for multimedia data streams, 1993. Preprint.

[18] F. MacWilliams and N. Sloane. *The Theory of Error Correcting Codes*. North-Holland, New York, 1977.

[19] N. Maxemchuk. Dispersity routing. In *Proc. ICC '75*, 1975.

[20] Anthony McAuley. Reliable broadband communication using a burst erasure correcting code. In *Proc. ACM SIGCOMM '90*, pages 197–306, 1990.

[21] H. Ohta and T. Kitami. A cell loss recovery method using FEC in ATM networks. *IEEE J. Select. Areas Commun.*, 9(9):1471–1483, 1991.

[22] K. Park, G. Kim, and M. Crovella. On the relationship between file sizes, transport protocols, and self-similar network traffic. In *Proc. IEEE International Conference on Network Protocols*, pages 171–180, October 1996.

[23] Kihong Park. Warp control: a dynamically stable congestion protocol and its analysis. *Journal of High Speed Networks*, 2(4):373–404, 1993.

[24] Lawrence Perko. *Differential Equations and Dynamical Systems*. Springer-Verlag, 1991.

[25] Michael Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM*, 36(2):335–348, 1989.

[26] Luigi Rizzo. Effective erasure codes for reliable computer communication protocols. *Computer Communication Review*, 27(2):24–36, 1997.

[27] B. Ryu and A. Elwalid. The importance of long-range dependence of VBR video traffic in ATM traffic engineering: myths and realities. In *Proc. ACM SIGCOMM '96*, pages 3–14, 1996.

[28] N. Shacham and P. McKenny. Packet recovery in high-speed networks using coding. In *Proc. IEEE INFOCOM '90*, pages 124–131, 1990.