A Retransmission Strategy for Real-Time Streaming Over Satellite in Blockage with Long Memory

Huan Yao
MIT Lincoln Laboratory
Lexington, MA, USA
yaohuan@ll.mit.edu

Yuval Kochman and Gregory W. Wornell
Research Laboratory of Electronics, MIT
Cambridge, MA, USA
{yuvalko,gww}@mit.edu

***Abstract*—The channel for a Comm-on-the-move (COTM) terminal in a blockage environment communicating over a satellite can be characterized as a packet erasure channel with long channel memory and long feedback delay. The goal of our research is to enable real-time application, such as a two-way voice call, over such a challenging channel. These packets need to be delivered reliably with strict order requirement. While various automatic repeat request (ARQ) techniques are often used for this purpose, they become ineffective when the channel is severely blocked with long memory and long feedback delay, and the user desires delay performance that is only a small amount above the physical limit of the channel while not consuming too much bandwidth.**

We propose a simple periodic retransmit solution for this problem. For example, for a 50% blocked channel with the round trip time being about the same as the channel memory duration, we can achieve average packet delay that is less than one channel memory duration over the minimum possible, while being more than 50% efficient with the bandwidth consumed.

For this simple periodic retransmit scheme, we analyze the expected delay of a packet, and compute the difference from the best possible delay. We evaluate the bandwidth consumption and show a delay-throughput tradeoff. We also suggest rules for selecting operating points that balances delay and throughput. Furthermore, we characterize user perception metrics of expected duration and frequency of playback interruptions. Simulations are used to validate the analysis.

***Index Terms*—real-time communication; communications-on-the-move (COTM); satellite; packet-loss channel; erasure; blockage; channel with memory; Markov model; delay; throughput; link-layer retransmission; ARQ;**

I. Introduction

The channel for a Comm-on-the-move (COTM) terminal in a blockage environment communicating over a satellite can be characterized as a packet erasure channel with long channel memory and long channel propagation and feedback delay. The goal of our research is to enable real-time streaming, such as video or voice, over such a challenging channel. The stream may be a long stream, such as a surveillance video for situational awareness, or short bursts, as in conversations. A fundamental limitation on the delay performance is due to the physical channel blockage, i.e., packets generated during channel blockage cannot get through while the channel is blocked and must wait for channel to open up. We would like to have a retransmission strategy that can achieve delay performance that is only a small amount above the best possible and yet not consume a large amount of bandwidth with unnecessarily frequent retransmissions.

Traditionally, ARQ methods have been used for reliable delivery of ordered packets over unreliable links. However, known ARQ methods may not work well for severely-blocked long-memory long-delay channels and may not work well for packets that are generated and consumed sequentially in real time with low desired delay. For examples, for an interactive voice application (one packet per 20 ms), users may desire a delay performance of no more than a few seconds, even when the channel delay and channel blockage memory are both on the order of a second and the channel is blocked 50% of the time.

Basic ARQ techniques, stop-and-wait ARQ, go-back-N ARQ, and selective-repeat ARQ, become inefficient as the channel loss rate becomes higher than 10^-3 [1]. Hybrid ARQ schemes that utilize forward error correction (FEC) coding were designed to reduce the effective channel loss rate [1]–[3]. When the channel degradation is mainly due to noise, FEC can be quite effective. However, for blockage channels, the FEC would have to span many (possibly tens) independent channel realizations to achieve the same effect. When the channel memory is long, the coding would induce a long delay and a high complexity. Without coding, the ARQ mechanism still needs to deal with the high packet loss rate. In [4], a hybrid ARQ technique was used to support interactive voice over a moderately blocked satellite channel as a COTM terminal was driven in a city environment, and the user experience was poor.

Furthermore, most ARQ schemes focus on the throughput achieved by the system rather than delay. In all ARQ schemes, including the hybrid ARQ schemes, retransmissions are only performed after a full round trip time (RTT). When the channel loss rate is high, there is a non-negligible probability that it would take several RTT for a packet to get through. When ordered packet delivery is required, one such packet would delay all packets that come after. For real-time applications, it is desirable to achieve delay that is only a few RTT or even less than one RTT. If the physical channel is blocked for multiple seconds, the delay should be just a little bit beyond that. To achieve this level of low delay over a severely-blocked

This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

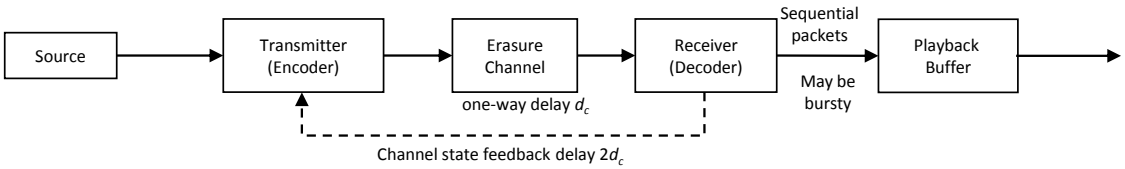

Fig. 1. Streaming system block diagram.

long-memory long-delay channel, a different kind of scheme is needed.

Our prior work [5]–[7] studied this problem without the channel memory consideration; the blockages were modeled as independent and identically distributed (IID) events. A multi-burst transmission scheme was proposed where a number of retransmissions are sent over independent channel realizations in one RTT, and the number increases as we get close to a desired delay target. The scheme has the option to yield low delay very close to the physical channel limit by using a large number of retransmissions. We showed that there is a fundamental tradeoff between the delay achieved and the bandwidth consumption; we could trade delay for bandwidth saving and vice versa.

This work addresses the scenario with long channel memory, which could last for many seconds [4]. In this case, the channel memory is longer than or on the same order as channel delay and the duration of some short messages. One possible scenario is where the channel memory, channel delay, and message length are all on the order of one to a few seconds.

In this paper, we propose a rather simple periodic retransmit scheme for the problem. It is likely that this is not the best that can be done, but we will show that it has reasonably good performance when the retransmission interval, δ, is chosen appropriately. Generally, δ that is too large would lead to delay much larger than the physical limit, and a δ that is too small relative to the channel memory would gain little in delay and yet incur a lot of bandwidth consumption.

This paper is outlined as follows. In Section II, the system model is given including the model for the channel memory. In Section III, we determine the expected delay of a packet and compute the difference from the best delay achievable by any scheme, which is only limited by the physical channel blockages. In Section IV, the bandwidth requirement is evaluated. Section V shows the tradeoff between the delay achieved and the bandwidth requirement, while Section VI suggest rules for selecting operating points depending on the channel blockage probability and how the channel delay compares with the channel memory duration. We show that it is indeed possible to achieve delay that is slightly above the physical limit with reasonable bandwidth consumption. For example, for a 50% blocked channel with the round trip time being about the same as the channel memory duration, we can achieve average packet delay that is less than one channel memory duration over the minimum possible, while being more than 50% efficient with the bandwidth consumed. In Section VII,

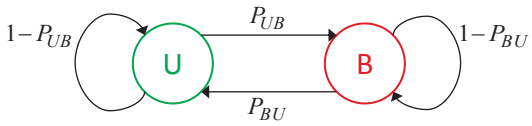

Fig. 2. Continuous-time two-state Markov model for channel erasures.

we analyze the a few user perception metrics such as expected duration and frequency of playback interruptions. Section VIII presents simulation results that validate the analysis. SectionIX provides some concluding remarks.

II. System Model

The system block diagram is shown in Fig. 1 It is very similar to the one used in [5]–[7], except modified to have a continuous source and a channel with memory.

A source generates a constant-rate stream of duration L from time 0 to time L. We assume the source symbol or packet interval is sufficient small, so the discrete effects are negligible. The transmitter transmits on integer multiples of a time step δ, [1] which is the one and only design parameter. Source data from time $(k-1)\delta$ to $k\delta$, $k = 1, 2, 3, \cdots, K$, where $K = \lceil L/\delta \rceil$, are bundled into a packet p_k. Each packet p_k is immediately transmitted at time $k\delta$, and repeatedly retransmitted at times $(k+1)\delta, (k+2)\delta, \cdots$, until an acknowledgment (ACK) for packet p_k is received. These (re)transmissions are carried out for all packets independently and simultaneously. For example, at time 3δ, if p_1 and p_2 have not be acknowledged, then p_1, p_2, and p_3 are transmitted together. One inherent assumption here is dynamic bandwidth consumption, also used in [7]. It enables channel adaptation and is feasible when the stream of interest shares resources with other traffic. [2]

The erasure channel with channel memory is modeled using a continuous-time two-state Markov model, as shown in Fig. 2.

Let U and B denote the channel open (unblocked) and blocked states. One common way to specify a continuous-time two-state Markov model is using the average duration of time spent in each state, i.e., T_U and T_B. An alternative way

[1] In practice, this may undesirably lead to bursty transmissions every δ. Instead, each packet should be transmitted as it arrives and retransmitted every δ on a packet by packet basis.

[2] Another assumption is unlimited peak bandwidth. In practice, it is sufficient to set the peak bandwidth to be slightly more than the average bandwidth consumption. [7]

is to use the channel open probability ρ and channel memory duration T_0, where

$$\rho = \frac{T_U}{T_U + T_B} \text{ and } T_0 = \frac{T_U T_B}{T_U + T_B}. \quad (1)$$

The steady state probability of being in each state is

$$P_U = \rho \text{ and } P_B = 1 - \rho. \quad (2)$$

Let $s(t)$ denote the channel state at time t. The state transition probabilities across time $\tau > 0$ are

$$P_{UU}(\tau) \triangleq \Pr\{s(t+\tau) = 1 | s(t) = 1\} = \rho + (1-\rho)e^{-\tau/T_0},$$
$$P_{BB}(\tau) \triangleq \Pr\{s(t+\tau) = 0 | s(t) = 0\} = 1 - \rho + \rho\ e^{-\tau/T_0},$$
$$P_{UB}(\tau) = 1 - P_{UU}(\tau), \text{ and } P_{BU}(\tau) = 1 - P_{BB}(\tau). \quad (3)$$

Any transmission made at time t reaches the receiver after a delay of d_c if $s(t) = 1$, or is blocked if $s(t) = 0$. The transmitter learns the channel state through an error-free packet acknowledgment fed back after a further delay of d_c. The effective round trip time (RTT), i.e., the time it takes for a packet loss to be responded by a new transmission, is $\text{RTT} = (\lfloor 2d_c/\delta \rfloor + 1)\delta$.

Even though the channel model is continuous-time, since the transmitter only transmits on integer multiples of δ, the effective channel is really a discrete-time one. However, as the time step δ is a variable, it is necessary to define the model in continuous-time terms to keep the discrete channel models consistent as δ changes.

As stated earlier, this study focuses on the case where the channel memory T_0 is longer or is of the same order as d_c. The message duration L could be longer than or shorter than T_0. In the sequel, we will often need to express time quantities relative to T_0, such as in (3). For notational convenience, we use $\bar{\cdot}$ to represent time quantities scaled by T_0. For example, $\overline{d_c} = d_c/T_0$ and $\overline{L} = L/T_0$.

The receiver processes the received packets, sorts them, and eliminates duplicate packets received. It passes the packets sequentially to the playback buffer. It also sends acknowledgments back to the transmitter over the error-free feedback channel.

The playback buffer simply plays back the received packets once per time step δ. In the event that there is no packet to playback, which might happen after a long blockage, there would be an interruption in the playback. When the channel opens again, there would typically be a burst of packets received simultaneously. The oldest (smallest index) of those packets is played back first, all later packets are buffered and played back sequentially. Consequently, after each packet is received, it cannot be played back until all the preceding packets that have been buffered are played back. Although this causes additional delay, one benefit is that a short blockage after a longer blockage had occurred does not cause an interruption in playback, as there are buffered packets to playback to help "ride over" the short blockage. One potential way to reduce delay is to allow playback to happen in a "catch up" mode, where the packets are played back slightly faster $(e.g., 10\%)$ than normal. While we do not assume such capability in this paper, it could be useful in practice.

III. Delay Analysis - From a Packet Perspective

We use a similar packet delay definition as in [5]–[7]. In particular, the delay experienced by packet p_k is

$$D_k \triangleq M_k - k\delta, \quad (4)$$

where M_k denotes the time p_k is played back and $k\delta$ is the time p_k is generated. For example, p_1 is generated at time δ and immediately transmitted. If $s(\delta) = 1$ (channel open), then p_1 is received at $d_c + \delta$ and playback finishes at time $d_c + 2\delta$, so $D_1 = d_c + 2\delta - \delta = d_c + \delta$. This is the minimum delay every packet must experience.

More generally, a packet p_k is generated at time $k\delta$, and is played back at time $d_c + (1 + k + B_k)\delta$, where B_k is the longest stretch of consecutive blockage experienced by any packet up to and including p_k. This was shown in [5]–[7]. The intuition is that if any packet p_j, $1 \le j \le k$ experiences B_k blockage consecutively, p_j itself suffers a delay of $d_c + (1 + B_k)\delta$. All packets after it must suffer at least this delay.

The average packet delay for p_k can be calculated using

$$E[D_k] = d_c + \delta + \delta \sum_{b=1}^{\infty} Pr\{B_k \ge b\}. \quad (5)$$

To evaluate $Pr\{B_k \ge b\}$, we see that in order to have b consecutive zeros, the channel sequence must take the form

$$\underbrace{0 \cdots 0}_{b} * \cdots *,$$

or

$$\underbrace{* \cdots *}_{i-1} \underbrace{1 0 \cdots 0}_{b} * \cdots *, \quad \text{for } i = 1, 2, \cdots, k-1.$$

The probability of having a row of b blockages at the beginning is $P_B P_{BB}(\delta)^{b-1}$. For each $i = 1, 2, \cdots, k-1$, the probability of having a particular pattern of this form is $P_U P_{UB}(\delta) P_{BB}(\delta)^{b-1}$. Using the union bound,

$$Pr\{B_k \ge b\} \le \min(1, P_B P_{BB}^{b-1} + (k-1)\, P_U P_{UB} P_{BB}^{b-1}. \quad (6)$$

Combining (3), (5) and (6), the expected delay of an entire stream of duration L, $D(L) = D_{K=\lceil L/\delta \rceil}$, can be upper bounded by

$$E[\overline{D(L)}] \le \overline{d_c} + \overline{\delta} + \overline{\delta} \sum_{b=1}^{\infty} \min\left(1, (1-\rho) \cdot \left(1 - \rho + \rho \cdot e^{-\overline{\delta}}\right)^{b-1} \left(1 + \left(\lceil \overline{L}/\overline{\delta} \rceil - 1\right) \rho (1 - e^{-\overline{\delta}})\right)\right) \quad (7)$$

The right hand side of (7) contains the channel parameters, ρ and $\overline{d_c}$, the stream duration $\overline{L}$, and the time step $\overline{\delta}$, which is a design parameter. Fig. 3 shows the excess delay above the channel delay, $E[\overline{D(L)}] - \overline{d_c}$, as a function of $\overline{L}$ in log scale, for $\rho = 0.5$ and $\delta = 0.01$, 0.1, 0.5, and 1.0.

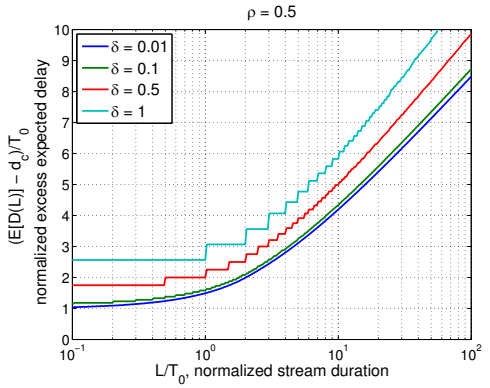

Fig. 3. Normalized excess expected delay $E[\overline{D(L)}] - \overline{d_c}$ as a function of normalized stream duration $\overline{L}$ for $\overline{\delta} = 0.01$, 0.1, 0.5, and 1.0.

Looking at one curve at a time, each curve starts off flat for $\overline{L} < 1$. This is because when the stream duration is shorter than the channel memory, the entire stream would essentially see the same channel, and it is the average channel blockage duration that dominates the delay. Indeed, the starting point is approximately $\frac{1-\rho}{\rho} + \frac{1+\rho}{2\rho}\overline{\delta}$ for $\overline{\delta} \leq 1$. [3] The first term is essentially $P_B\overline{T_B}$, the probability that the stream sees a blocked channel at onset times the expected blockage duration.

For large $\overline{L}$, delay grows linearly with $\log(\overline{L})$, similar to the conclusion in [5]–[7]. This is because with a two-state Markov model, the blockage duration is exponentially distributed, so statistically, it would take an exponentially longer stream to hit a blockage that is a fixed amount longer. The asymptotic slope of each curve with respect to $\ln(\overline{L})$ is approximately $\frac{1}{\rho} + \frac{1-\rho}{2\rho}\overline{\delta}$, for $\overline{\delta} < 1$. [4] [5]

Comparing the different curves, lower delays are achieved for smaller values of δ. In the limit of $\delta \to 0$, the transmitter essentially tries to transmit continuously (unrealistic, as it would consume a prohibitively large amount of bandwidth), so every packet is put through the channel as soon as the channel opens after the source generates it, and thus achieving the minimum delay physically possible. The bottom curve with $\overline{\delta} = 0.01$ essentially corresponds to this limit. Compared to this bottom curve, the delay gradually increases as δ increases. $\overline{\delta} = 0.1$ has a very small gap to the minimum delay. $\overline{\delta} = 0.5$ suffers additional delay of less than 0.9 for $\overline{L} < 10$; even at $\overline{L} = 100$, the additional delay is less than 1.4, or just 1.4% of the stream length. For $\overline{L} > 100$, the gap grows at a rate of 0.65 per decade. $\overline{\delta} = 1.0$ suffers about twice as much delay degradation as $\overline{\delta} = 0.5$. The steps in the curves are due to the $\lceil \overline{L}/\overline{\delta} \rceil$ term in (7).

Note that packet delay is independent of the channel RTT, since the retransmission interval is δ rather than RTT used in most traditional ARQ schemes. Also, for different values of ρ, such as 0.8 and 0.2, Fig. 3 retains its shape except the delay scale is changed. More delay is expected for more severe blockage.

Next, we define a delay metric (DM) that captures the differences in delay. Let $D^{min}(L)$ be the minimum packet delay achieved with $\delta = 0$. This represents the physical limit of the channel. We define

$$\mathrm{DM} = \lim_{L \to 0} \overline{D(L)} - \overline{D^{\min}(L)}. \tag{8}$$

It is defined for $L \to 0$ so it is independent of L. As shown in Fig. 3, when L is small, the difference is about the same as at $L \to 0$; when L is large, although the gap grows, it grows logarithmically at a slow rate. In the particular example of $\rho = 0.5$ and $\overline{\delta} = 0.5$, the rate was just 0.65 per decade.

IV. Throughput Metric

To capture the bandwidth consumption, we compute the throughput metric, originally defined in [5]–[7]. In this setup, the throughput metric TM is the ratio between the expected total number of packets received and the total number of packets that need to be sent. As packet retransmissions are individually handled, this is equivalent to the expected number of times each packet is received by the receiver. $1/\mathrm{TM}$ is essentially bandwidth efficiency. The minimum TM is 1, as each packet must be received at least once. As there may be unnecessary retransmission due to the long feedback delay, TM could be much larger than 1 leading to low efficiency. For example, if each packet is expected to be received 5 times on average, the efficiency is 20%. The reason we define TM based on reception rather than transmission is that the minimum TM would be 1 independent of channel blockage, even when the channel is severely blocked.

To compute TM, we see that after each packet is first successfully transmitted through the channel, while waiting for the ACK, the transmitter will continue transmission for $\overline{\mathrm{RTT}}/\overline{\delta} - 1$ extra times. Therefore,

$$\begin{aligned} TM &= 1 + \sum_{i=1}^{\overline{\mathrm{RTT}}/\overline{\delta}-1} \left(\rho + (1-\rho)e^{-i\cdot\overline{\delta}}\right) \\ &= \sum_{i=0}^{\overline{\mathrm{RTT}}/\overline{\delta}-1} \left(\rho + (1-\rho)e^{-i\cdot\overline{\delta}}\right) \\ &= \rho \cdot \frac{\overline{\mathrm{RTT}}}{\overline{\delta}} + (1-\rho)\cdot\frac{1-e^{-\overline{\mathrm{RTT}}}}{1-e^{-\overline{\delta}}} \end{aligned} \tag{9}$$

When $\delta = \mathrm{RTT}$, the scheme becomes standard ARQ, where the transmitter waits for a full RTT to ensure that the packet is indeed lost before retransmission, and the minimum TM of 1 is achieved. In (9), we see that the first term contains $1/\overline{\delta}$. [6] When $\overline{\delta}$ is small, TM would grow like $1/\overline{\delta}$. Therefore, very small $\overline{\delta}$ should be avoided.

[3] The exact starting point is $\frac{1-\rho e^{-\overline{\delta}}}{\rho(1-e^{-\overline{\delta}})}\overline{\delta} \approx \frac{1-\rho}{\rho} + \frac{1+\rho}{2\rho}\overline{\delta} + \frac{1-\rho}{12\rho}\overline{\delta}^2 + \cdots$.

[4] The exact slope with respect to $\ln(\overline{L})$ is $\frac{-\overline{\delta}}{\ln(1-\rho+\rho e^{-\overline{\delta}})}$.

[5] The knee of the curve is at $\frac{e^{-\rho}}{\rho(1-\rho)}$ for $\overline{\delta} = 0$.

[6] In the second term, $1 - e^{-\overline{\delta}} \approx \overline{\delta}$ when $\overline{\delta}$ is small. So effectively, the second term also contains $1/\overline{\delta}$.

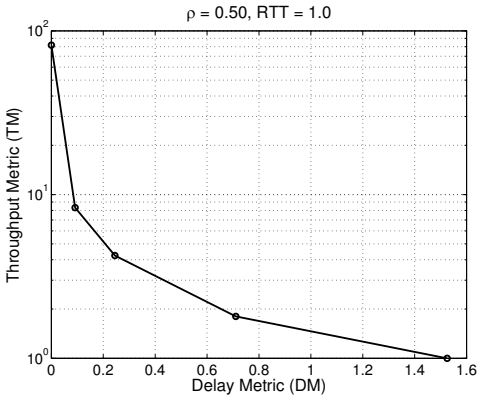

Fig. 4. The delay-throughput tradeoff achieved using $\bar{\delta} = 0.01$, 0.1, 0.2, 0.5, and 1.0 for the case of $\rho = 0.5$ and $\overline{\mathrm{RTT}} = 1.0$.

V. Delay-Throughput Tradeoff

In Section III, we showed that packet delay suffers some degradation with increasing δ. In Section IV, we showed that bandwidth utilization, i.e., the throughput metric, could be very large when δ is made too small. In this section, we study the tradeoff between the delay achievable and the bandwidth required.

Fig. 4 shows the DM-TM tradeoff for the case of $\rho = 0.5$ and $\overline{\mathrm{RTT}} = 1$. The curve shows the various DM/TM pairs achieved as $\bar{\delta}$ takes the values of 0.01, 0.1, 0.2, 0.5, and 1.0. [7] The last point with $\bar{\delta} = 1.0 = \overline{\mathrm{RTT}}$ achieves the minimum TM of 1. The second last point with $\bar{\delta} = 0.5 = \rho$ achieves a DM of 0.71 and a TM of 1.8, which corresponds to delay less than T_0 over the physical minimum possible and about $\sim 55\%$ bandwidth utilization efficiency.

Another way to get some intuition on the delay-throughput tradeoff is through looking at approximate expressions of DM and TM instead of the exact expressions. For DM, we have (see second paragraph below (7))

$$\mathrm{DM} \approx \frac{1+\rho}{2\rho}\bar{\delta} \leq \frac{\bar{\delta}}{\rho}. \tag{10}$$

For TM, the worst that could happen is that all $\overline{\mathrm{RTT}}/\bar{\delta} - 1$ retransmissions are received, so

$$\rho \cdot \frac{\overline{\mathrm{RTT}}}{\bar{\delta}} \leq \mathrm{TM} \leq \frac{\overline{\mathrm{RTT}}}{\bar{\delta}}. \tag{11}$$

So DM is proportional to $\bar{\delta}$ and TM is inversely proportional to $\bar{\delta}$. The smaller the $\bar{\delta}$, the more frequent the retransmissions, the better delay but higher bandwidth consumption. The product is dictated by $\frac{\overline{\mathrm{RTT}}}{\rho}$, the channel statistics. While this is not exact, it provides a good intuition.

VI. Retransmit Interval Selection

In the last section, we studied the delay-throughput tradeoff. In this section, we provide some rule-of-thumb recommendation on how to select the retransmit interval δ for different scenarios.

1) δ should never be more than RTT. The minimum TM of 1 is achieved when $\delta = \mathrm{RTT}$, increasing δ further would only hurt delay.
2) When $\overline{\mathrm{RTT}} << 1$, use $\delta = \mathrm{RTT}$. This is essentially the ideal selective-repeat ARQ. [8]
3) When $\overline{\mathrm{RTT}} \approx 1$, for light to moderate blockage with $0.5 \leq \rho \leq 1$, use $\bar{\delta} = \rho$. This achieves DM between 0.75 and 1 and achieves TM between $\overline{\mathrm{RTT}}$ and $2\overline{\mathrm{RTT}}$.
4) When $\overline{\mathrm{RTT}} \approx 1$ and severe blockage with $\rho < 0.5$, use $\bar{\delta} = 0.5$ if willing to suffer delay from $0.25/\rho$ to $0.5/\rho$; use $\bar{\delta} = \rho$ if willing to pay bandwidth up to $\overline{\mathrm{RTT}}/\rho$.
5) When $\overline{\mathrm{RTT}} >> 1$, use time step $\bar{\delta}$ between 1 and 2, so the transmissions see nearly independent channel realizations. Also, instead of retransmit every δ, do transmission only a few times per RTT according to the multi-burst transmission strategy in [5]–[7].

In practice, we may not have accurate knowledge of the channel parameters ρ, d_c, and T_0. In this case, we could utilize the approximations of DM and TM in (10) and (11) and select δ that leads to acceptable DM and TM given the channel parameter ranges.

Another general rule is that when the primary performance metric is low delay, the retransmission interval should scale with channel memory; while if the the primary performance metric is high throughput efficiency, the retransmission interval should scale with the channel RTT. Indeed, in traditional ARQ techniques, the primary performance metric is high throughput efficiency, and retransmission typically occur once every RTT. In the scenario of interest in this paper, we want to support real-time applications with low delay requirement, retransmission interval is often tied to the channel memory, so we select $\bar{\delta}$, which is essentially how long the retransmission interval is relative to the channel memory.

VII. User Perspective Analysis

While this paper establishes the analytical results, we are currently also working on an emulation [9] where two telephones are connected via a PC emulating the severely-blocked high-delay long-memory channel and running the proposed algorithm. As our initial testing has revealed, in addition to packet delay, playback interruptions also strongly impact the user experience. These playback interruptions occur most often early in the transmission. In this section, we analyze how long the interruptions are and how frequently they occur.

To analyze the above quantities, we define a slightly different variant of delay. Let the user perceived delay, $U(t)$ be the difference between how much has been generated and how much has been played back at time t. When t is an integer multiple of δ, we have

$$U_k \triangleq U(k\delta) = k\delta - j\delta, \text{ for } k = 1, 2, \cdots, K, \tag{12}$$

[7] While the DM is defined with $L \to 0$, for larger values of L such as 100 and 1000, we saw that the shape of the curve remains the same with the delay axis scaled by a factor of 2.2 and 3.0, respectively.

[8] If one were interested in reducing the number of transmissions, one could potentially hold off retransmissions until there is an indication that the channel has opened up.

[9] Developed by Dr. Mehmet Mustafa at MIT Lincoln Laboratory.

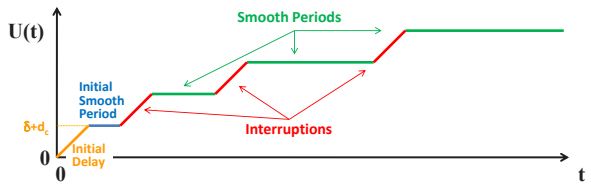

Fig. 5. An example of user perceived delay showing interruptions and increasing smooth playbacks after an initial delay and possible smooth period.

where $j < k$ is the most recent packet played back or 0 if no packet has been played back at all. Values of $U(t)$ for t between integer multiples of δ can be obtained via linear interpolation, as we assume packet generating and playback both happen continuously at constant rate.

Fig. 5 shows an example of $U(t)$. It always start at $U(0) = 0$ and initially climbs to $U(\delta + d_c) = \delta + d_c$ (orange), as nothing can be played back for at least that long. If the channel is initially open, there would be an initial smooth period (blue); if the channel is initially blocked, this period would have zero duration. After that, Each time there is an interruption in playback, $U(t)$ would climb (red), and whenever there is smooth playback, $U(t)$ stays flat (green). We are interested in the expected duration of each interruption period and each smooth period.

Each interruption occurs when a blockage duration exceeds the longest blockage that had occurred previously, and ends when the channel opens up again. Since the channel is memoryless, independent of how long a blockage has been, the expected remaining blockage is always

$$E[\text{interruption duration}] = \frac{\delta}{P_{BU}(\delta)} = \frac{1}{\rho} \cdot \frac{\delta}{1 - e^{-\bar{\delta}}} = T_B \cdot \frac{\bar{\delta}}{1 - e^{-\bar{\delta}}}. \tag{13}$$

Similarly, each time the channel becomes open, the expected open duration is $\frac{\delta}{P_{UB}}$. Since there is a chance of ρ that the channel is initially open, the expected initial smooth period is

$$E[\text{initial smooth duration}] = \rho \cdot \frac{\delta}{P_{UB}(\delta)} + (1 - \rho) \cdot 0 = \rho \cdot T_U \cdot \frac{\bar{\delta}}{1 - e^{-\bar{\delta}}}. \tag{14}$$

After the initial smooth period, the duration of each smooth period is the sum of a number of blockage and open periods. The number, denoted by N, is random, and depends on how many blockages takes place before there is one that is longer than all the previous blockages. The probability that one blockage lasts (strictly) longer than $Z\delta$, $Z = 1, 2, 3, \cdots$, is $P_{BB}(\delta)^Z$. Therefore, $E[N] = P_{BB}(\delta)^{-Z}$. When N blockages are required before an interruption occurs, the length of the smooth period is the sum of N open periods, each expected to be $\frac{\delta}{1-P_{UU}(\delta)}$ long, $N - 1$ blockage periods that are all no longer than $Z\delta$, and a final blockage period that is exactly $Z\delta$ long, during which the playback buffer is depleted. Therefore, when the user perceived delay U is $\delta + d_c + Z\delta$, which is the delay resulted from the longest blockage being $Z\delta$, $Z \geq 1$,

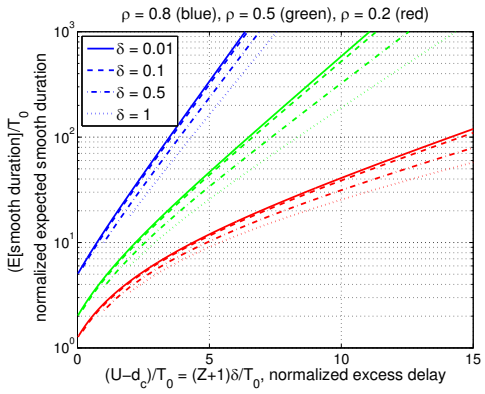

Fig. 6. Normalized expected smooth duration $\overline{E[\text{smooth duration}]}$ as a function of normalized excess delay $\overline{U - d_c} = (Z + 1)\bar{\delta}$ for $\rho = 0.8$, 0.5, and 0.2 (different colors), and $\bar{\delta} = 0.01$, 0.1, 0.5, and 1.0 (different line types).

the expected smooth duration is

$$\begin{aligned} &E[\text{smooth duration at } U = \delta + d_c + Z\delta, Z = 1, 2, 3, \cdots] \\ &= E[N] \cdot E[\text{one open period}] + \\ &(E[N] - 1) \cdot E[\text{one blockage period} \mid \text{it is } \leq Z\delta] + Z\delta \\ &= \frac{\delta}{1 - e^{-\bar{\delta}}} \left(\frac{(1 - \rho + \rho e^{-\bar{\delta}})^{-Z}}{\rho(1 - \rho)} - \frac{1}{\rho} \right) \qquad (15) \\ &= T_U \cdot \frac{\bar{\delta}}{1 - e^{-\bar{\delta}}} \cdot \frac{(1 - \rho + \rho e^{-\bar{\delta}})^{-Z} + \rho - 1}{\rho} \qquad (16) \end{aligned}$$

Fig. 6 shows the normalized expected smooth duration as a function of the normalized excess delay, $\overline{U - d_c} = (Z + 1)\bar{\delta}$, for various values of ρ and $\bar{\delta}$. The y-axis is on log scale, which means that as delay increases, the smooth duration increases exponentially. We see that for each ρ, the better performance, longer smooth duration, is achieved with smaller δ. Similar to what we saw with the packet based delay, $\delta = 0.1$ is nearly as good as $\delta = 0.01$, and the performance degradation of $\delta = 1.0$ is about twice that of $\delta = 0.5$. When $\rho = 0.5$, a smooth period of 100 can be achieved with excess delay of about 7. Since each interruption tends to last about $1/\rho = 2$. it would take about 3 or 4 interruptions before we can get such long smooth playback. When $\rho = 0.8$, the blockage is relatively light, a smooth period of 100 can be achieved with excess delay of just 4, which also corresponds to 3 to 4 interruptions. When $\rho = 0.2$, the blockage is heavy, a delay of 15 is required, which is $3/\rho$. Generally, the rule of thumb is that after one interruption, the delay and smooth period are about $1/\rho$ and 10, after two interruptions, $2/\rho$ and 30, after three interruptions, $3/\rho$ and 100, all in the unit of channel memory time T_0.

VIII. Simulation Results

This section shows simulation results to verify the analytical results. The channel parameter is $\rho = 0.5$, $T_0 = 1$ sec, and $d_c = 1$ sec. For this channel, the average open and blockage durations are $T_U = T_B = 2$. Three values of δ are used, 0.1 sec for achieving near minimum delay at the expense of bandwidth, and 0.5 seconds and 1.0 seconds for more practical

	$\delta = 0.1$ sec		$\delta = 0.5$ sec		$\delta = 1.0$ sec	
	Simulation	Analytical	Simulation	Analytical	Simulation	Analytical
Throughput Metric (TM)	15.12	15.11	3.668	3.666	2.252	2.252
$E[D(L)]$, $L = 1$ sec	2.546	2.581	3.034	3.002	3.603	3.565
$E[D(L)]$, $L = 100$ sec	9.020	9.723	10.14	10.86	11.56	12.43
E[interruption duration]	2.095	2.102	2.544	2.541	3.163	3.164
E[smooth duration], $U = 2 + d_c$	9.048	9.042	9.740	9.668	10.77	10.36
E[smooth duration], $U = 4 + d_c$	26.55	27.44	26.68	26.78	25.11	25.76

TABLE I
COMPARISON OF SIMULATION AND ANALYTICAL RESULTS FOR THE CASE $\rho = 0.5$, $T_0 = 1$ SEC, AND $d_c = 1$ SEC

operation. For each setting, over $10,000$ trials are run so that the error on the measured quantities are less than 1%. The results are shown in Table I.

Table I shows that for the throughput metric, the simulation and analytical results match nearly perfectly. When $\bar{\delta} = 0.1$, TM is very large; when $\bar{\delta}$ increases to 0.5, TM is reduced significantly; $\bar{\delta} = 1.0$ further reduces TM, but the gain is diminishing.

The second and third rows show the expected delay for short $L = 1$ sec and long $L = 100$ sec messages. Delay increases with δ as expected. However, even at $\bar{\delta} = 0.1$ with delay close to the physical channel limit, the delay is already quite large. For the short message case with $L = 1$, the delay is dominated by $d_c + \frac{1-\rho}{rho} T_0 = 2$ sec; for the $L = 100$ sec case, the delay is about 9 sec. Compare to $\bar{\delta} = 1.0$, even though $\bar{\delta} = 0.5$ can achieve delay that is about 50% closer the minimum possible, the absolute difference is small. This behavior can also be seen in Fig. 3. Compare to the analytical results, the simulation results are slightly less, because the union bound used in calculating the analytical results leads to over-estimation.

The fourth row shows the expected interruption duration. Each duration only lasts 2 to 3 seconds. This would certainly be noticeable by the users. However, as shown in (13), this is essentially dominated by T_B, which is the physically limit of the channel. Using $\delta = 0.5$ and 1.0 causes the interruption to be about 30% and 60% longer, respectively.

The last two rows shows the expected smooth duration when the user perceived delay U is 2 seconds and 4 seconds above the minimum delay due to channel propagation. As δ increases, the expected smooth duration decreases slightly. But nominally, after having experienced 2 seconds of interruption, the user can expect 10 seconds of smooth playback; after 4 seconds of interruption, the expected smooth duration is nearly half a minute.

Another way of viewing this result is that if a user in such a scenario were to limit the maximum perceived delay to $4 + d_c = 5$ seconds, and skip packets that are not received in time for playback, then this user can expect to have smooth playbacks of half minutes long with interruptions that last 2 to 3 seconds. This translates to 6% to 10% of (bursty) packet losses.

Finally, in both analysis and simulation, it is assumed that acknowledgments are perfectly received. This may be mitigated by sending acknowledgments that cumulatively acknowledges a range of packets, which has been implemented in emulation.

IX. SUMMARY

We studied the problem of real-time streaming over blockage channel with long feedback delay and long channel memory. We showed that most time quantities scales with the channel memory duration T_0, including the desired time step size δ, the packet delay, as well as the interruption durations and smooth playback durations. We showed that when stream length is shorter than channel memory duration, packet delay does not change much; when stream length gets longer, packet delay grows logarithmically. We showed that we can expect reasonably long smooth playback periods after suffering a few interruptions. We evaluated the delay-throughput tradeoff and made suggestions on how to pick the retransmission interval $\bar{\delta}$ to achieve reasonably good delay while not consuming too much bandwidth. The general rule is that when $RTT << T_0$, simply use standard ARQ with $\delta = \mathrm{RTT}$, wait a full RTT before retransmission; when $RTT >> T_0$, choose time step $\delta = 2T_0$, so that the channel uses are essentially experience IID blockage, and use multi-burst transmission in [5]–[7]; when $RTT \approx T_0$, using $\delta = \rho \cdot T_0$ achieves TM of up to RTT/ρ and delay that is only about one channel memory duration above the physical limit. However, the minimum delay due to physical limit could be quite large itself.

REFERENCES

[1] S. Lin, D. Costello, "Automatic-repeat-request error-control schemes," IEEE Communications Magazine, vol. 22, pp. 5-17, Dec 1984.

[2] L. Shu, P. Yu, "A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels," IEEE Trans. Comm. vol. 30, pp. 1701-1719, Jul 1982.

[3] D. M. Mandelbaum, "Adaptive-feedback coding scheme using incremental redundancy," IEEE Trans. Inform. Theory, vol. IT-20, pp. 388-389, May 1974.

[4] J. Schodorf, "EHF Satellite Communications on the Move: Experimental Results," Tech. Rep. 1087, MIT Lincoln Laboratory, Lexington, MA, Aug. 2003.

[5] H. Yao, Y. Kochman, G. W. Wornell, "Delay-Throughput Tradeoff for Streaming Over Blockage Channels with Delayed Feedback," in *MILCOM*, (San Jose, CA), Nov. 2010.

[6] H. Yao, Y. Kochman, G. W. Wornell, "On Delay in Real-Time Streaming Communication Systems, " in *Proc. Allerton Conf. Commun., Contr., Computing*, (Monticello, IL), Sep. 2010.

[7] H. Yao, Y. Kochman, G. W. Wornell, "A Multi-Burst Transmission Strategy for Streaming over Blockage Channels with Long Feedback Delay, " to appear in IEEE Journal on Selected Areas in Communications (JSAC), Advances in Military Networking and Communications, December, 2011.