Submitted to *IEEE Trans. Automat. Contr.*

A Universal Approach to Queuing with Distortion Control

Stark C. Draper, Mitchell D. Trott, and Gregory W. Wornell

Submitted August 2003

Abstract

An efficient buffer management algorithm is developed for queues that handle distortion-tolerant data under finite memory limitations. When multiresolution source codes are used, overflows are avoided, and significant performance gains realized, by progressively reducing the fidelity of signal descriptions in a controlled manner. The proposed approach is universal, i.e., it works without knowledge of queue arrival and departure statistics. More strongly, we show that its performance is sample-path optimal, i.e., it achieves an average distortion equal to the best achievable by any algorithm, including those designed with full non-causal knowledge of queue arrival and service times.

Index Terms — congestion control, buffer management, queuing, networks, multimedia, multiresolution, successive refinement, source coding, transcoding

1 Introduction

If arrivals to a finite-memory queue outpace departures over a span of time, the queue will overflow. The ensuing uncontrolled data loss can seriously reduce system performance. If, however, the queue buffers distortion-tolerant data such as audio, video, or images, it can use this characteristic to adjust fidelity as new signals arrive. In this paper we show how to exploit distortion-tolerance to lower signal fidelity in a controlled manner, thereby freeing memory resources, avoiding overflows, and increasing end-to-end fidelity in a dynamic fashion. The resulting buffer control mechanism implements a type of congestion control that takes into account the relative value of enqueued signal information.

We use multiresolution source codes (e.g., see [4, 14] and the references therein) to exploit the inherent distortion-tolerance of signals. These codes prioritize source information from most significant to least

This work has been supported in part by the National Science Foundation under Grant No. CCR-0073520, by MARCO/DARPA Contract No. 2001-CT-888, by Hewlett-Packard through the MIT/HP Alliance, and by Texas Instruments through the Leadership Universities Program. This work was presented in part at the International Conference on Acoustics, Speech, and Signal Processing, Orlando, FL, May 2002.

S. C. Draper was with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. He is now with the Edward S. Rogers Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4 (E-mail: `sdraper@comm.utoronto.ca`).

M. D. Trott is with Hewlett-Packard Laboratories, Palo Alto, CA 94304 (E-Mail: `trott@hpl.hp.com`).

G. W. Wornell is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 (E-mail: `gww@mit.edu`).

significant. Such structure allows the source to be reconstructed progressively as the code stream becomes available at the queue output. In a complementary manner, at the queue input, least significant information can be deleted first to free memory resources, leaving more significant information unperturbed.

The ordered data structure of multiresolution codes lends itself naturally to a pair of storage and transmission algorithms. On the one hand, if the buffer is close to overflow, least significant information can be deleted to free memory space. Conversely, most significant information can be transmitted first to guarantee it is not lost in future overflows. We formalize these intuitive ideas and show they form the basis for an optimal approach to buffer management. Regardless of the times of arrivals and departures, no algorithm can attain a lower average distortion. This means that the optimality of this algorithm is universal in that its optimality is not a function of source statistics.

In order to help understand the performance gains effected by distortion control, we also develop a "baseline" algorithm that treats all data as distortion intolerant. We compare the performances of the two algorithms under a simple model of queue arrivals and departures. Under this model, the performance of the universal algorithm closely approaches a simple bound that is not a function of algorithm design or memory size. While the use of multiresolution coding for distortion-tolerant data distribution has been proposed before, e.g., see [12], the combination with a queuing model, the priority queuing protocols, the bound on performance, and the sample-path optimality of the universal algorithm are all new.

As one application of these ideas, consider a wired-to-wireless gateway manager routing multimedia content as in Fig. 1. If the wired half of the infrastructure is of much higher capacity, any communications bottleneck will likely occur at the gateway. Because system demand can be unpredictable and time varying (e.g., heavier loads during the day than at night), protocols such as those designed herein that can adapt to a changing system load in real-time, and without forward planning, are particularly well suited. A second application is to image storage in, e.g., a digital camera. The first pictures taken can be stored in memory (i.e., the buffer) at high resolution. As more pictures are taken, the resolution of all images can be progressively lowered in parallel to fit the new data. The algorithms we develop give a structured way to manage this resolution/number-of-images tradeoff. We can also prioritize pictures so that favored pictures experience slower (or even no) lowering of resolution. Such a system could prove useful in autonomous sensor vehicle such as a submarine or interplanetary probe. System unpredictability may now exist in both input and output processes: in the input process because the vehicle does not know a priori when it will observe phenomena of interest, and in the output process because of communication rate variations caused by changing environmental conditions.

To contextualize this work, we discuss a number of distinct research areas related to the source coding and buffering situations that we consider. A quantization and queuing model similar to ours is used by a number

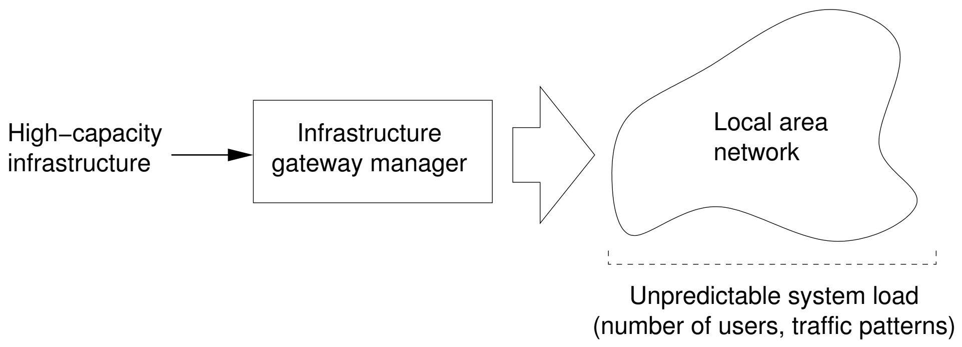
High−capacity infrastructure
Infrastructure gateway manager
Local area network
Unpredictable system load (number of users, traffic patterns)

Figure 1: Possible applications of buffer management algorithms include infrastructure gateway managers. The universal algorithm adjusts to match unpredictable traffic patterns.

of researchers [5, 8, 15, 16] who consider the effects of a variable-length source coder feeding a finite-memory queue that transmits the buffered information over a fixed-rate channel. These authors propose controlling the quantization rate based on the state of the queue to avoid overflows and to minimize average distortion. Our work differs from theirs because, by using multiresolution source codes, we effectively can change the quantization rate long after the source is quantized by deleting least significant information as a function of the state of the queue. Furthermore, we consider a queue buffering multiple data streams, each competing for resources, and our algorithms work across the different streams.

Multiple data streams feeding into finite-memory queues raises issues of network congestion. Our work is thus somewhat related to “active queue management” congestion control for packetized networks. Some standard approaches in this field, e.g., see [7, 6, 9] and the references therein, consider dropping or marking packets at intermediate nodes to allow end-to-end (source-to-destination) control to head off incipient network congestion. Our approach differs from these in part because we do not drop packets randomly. Rather, packet dropping is based on the relative importance of the information in the enqueued packets. Our approach is also more akin to node-by-node congestion control, though it can be used with end-to-end control by assigning each layer of resolution a different priority. The algorithms we develop are further differentiated by their use of prioritized transmission from the queue.

Finally, distortion controlled queuing can be combined in a complimentary manner with earlier work on multiple descriptions coding to give prioritized congestion control in unreliable networks. Multiple description codes are similar to multiresolution codes in that the source can be reconstructed progressively as more descriptions become available. However, they have an unordered data structure in that progressive reconstruction is possible for every order of description arrival. This makes multiple descriptions a particularly appropriate coding paradigm for unreliable networks, e.g., see [2, 11, 1, 3]. Distortion controlled queuing is easily used to prioritize information streaming in this context. Enqueued signals are still compared to each other to determine least significant information, but when descriptions pertaining to a particular signal are to be dropped or transmitted, it does not matter which. Note that the reliability gained through the unordered data structure of multiple descriptions comes at a cost. Multiple description codes require higher rates of communication to achieve the same distortion than do multiresolution codes, which have an ordered data structure. Therefore, in reliable networks, or in ones where node-by-node congestion control is used, ordered multiresolution coding is more efficient. We illustrate the ideas of this paper in this context.

The paper is organized as follows. Section 2 describes the system model and develops a simple bound on the performance of any algorithm. Section 3 presents the two algorithms for controlling queue memory: baseline and universal. While the universal algorithm is derived by optimizing a local cost function, and is therefore "greedy," we establish its globally optimality in Appendix A. Section 4 compares the algorithms for a simple process model and concludes the paper.

2 System Model and Performance Bound

In Fig. 2 we depict the system model. Signals arrive at the queue, are stored for some time, and then are sent on to their destinations (one or many decoders) over a shared packetized link. Three aspects of the model are important to note. First, each arrival is a 'full-resolution' signal in that all information relevant to that signal arrives concurrently. Second, each packet departs over a link shared by all signal destinations. Finally, our design objective is to minimize average decoder distortion.

The full-resolution arrival model is well matched to many network situations. In the infrastructure gateway example, it is a good model when the majority of system delay occurs at the gateway. In the autonomous vehicle example, arrivals are continuous sensor observations that must be quantized before being stored in memory. The shared link model allows information about different signals to be concatenated into a single packet. It is appropriate for broadcast systems such as cellular networks; scenarios where there is a single destination for all data, as in the sensor vehicle example; or when data streams are multiplexed for transmission along a sequence of network links. Finally, while the objective of minimizing average (rather than, e.g., peak) decoder distortion is useful for many distortion-tolerant data streams, it may be less well matched to others, such as video.

In the interval $[0, T]$ there are N_{arr} signal arrivals where the ith signal to arrive is denoted $\mathbf{s}_i$. Let B_i denote the number of bits describing $\mathbf{s}_i$ transmitted to its destination in this interval. The distortion in $\mathbf{s}_i$ at the end of the interval is given by the distortion-rate trade off $D[B_i]$. The maximum distortion, incurred when the decoder has no information about a signal, is $d_{\max} \equiv D[0]$. We denote the average distortion at time T

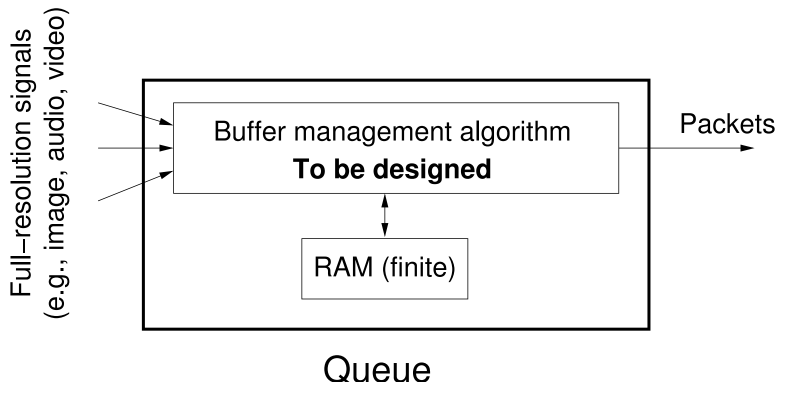

Figure 2: The universal algorithm capitalizes on distortion-tolerance to manage the buffering and distribution of data. Signals arrive at full resolution and departures are packets of a fixed size.

as d_T where

$$d_T = \frac{1}{N_{\text{arr}}} \sum_{i=1}^{N_{\text{arr}}} D[B_i]. \tag{1}$$

This is the function we minimize, ideally for all T.

We assume that $D[B_i]$ is convex and bounded (hence monotonic and continuous) and—to simplify derivations—differentiable.[1] This type of distortion function is appropriate, e.g., for the distortion-rate coding of random source sequences, for which it would define an expected distortion. In order to ease the presentation of key ideas, we focus on a common distortion measure for all $\mathbf{s}_i$, and typically treat bits as infinitely divisible, thereby avoiding integer constraints in the optimization problems. In Section 5 and Appendix A we comment on generalizations to signal-specific distortion measures.

In the interval $[0, T]$ there are also N_{dep} departures (i.e., transmission or service opportunities). Each departure is a packet of up to P bits. Putting this together with the N_{arr} arrivals gives the inequality

$$\sum_{i=1}^{N_{\text{arr}}} B_i \leq P N_{\text{dep}}, \tag{2}$$

where equality is achieved, e.g., if the queue never empties once the first signal arrives.

[1]The results of the paper could equivalently be developed in terms of the concave utility function $d_{\max} - D[B_i]$. We choose to develop the results in terms of distortion to highlight the potential benefits of accommodating interaction between source coding (typically an application-layer function) and congestion control (typically a transport or network-layer function).

We now derive a lower bound on d_T that is independent of memory size:

$$\begin{aligned} d_T &= \frac{1}{N_{\text{arr}}} \sum_{i=1}^{N_{\text{arr}}} D[B_i] \\ &\geq D\left[\frac{1}{N_{\text{arr}}} \sum_{i=1}^{N_{\text{arr}}} B_i\right] \qquad (3) \\ &\geq D\left[P \frac{N_{\text{dep}}}{N_{\text{arr}}}\right], \qquad (4) \end{aligned}$$

where (3) follows from Jensen's inequality, and (4) follows from (2). Assuming the limits exist, define $\lambda = \lim_{T\to\infty} N_{\text{arr}}/T$ to be the average arrival rate, $\mu = \lim_{T\to\infty} N_{\text{dep}}/T$ to be the average departure rate, and $\rho = \lambda/\mu$ to be the system utilization. A simple lower bound on the long term average distortion $d = \lim_{T\to\infty} d_T$ follows. Since (4) holds for all T,

$$\begin{aligned} d &\geq \lim_{T\to\infty} D\left[P \frac{N_{\text{dep}}}{N_{\text{arr}}}\right] \\ &= D\left[\lim_{T\to\infty} P \frac{N_{\text{dep}}}{N_{\text{arr}}}\right] \qquad (5) \\ &= D\left[P \lim_{T\to\infty} \frac{N_{\text{dep}}}{T} \lim_{T\to\infty} \frac{T}{N_{\text{arr}}}\right] \qquad (6) \\ &= D[P\mu/\lambda] = D[P/\rho]. \qquad (7) \end{aligned}$$

In (5) we use the fact that $D[\cdot]$ is continuous since it is convex and bounded, in (6) we can take the limits individually since both exist by assumption, and in (7) we use the definitions of μ, λ, and ρ.

The performance bound (7) tells us the ideal balance of description rates among signals. A good design minimizes the variance among description rates while maximizing individual description rates. If all signals are described at P/ρ, the average system throughput in bits per signal, then the bound (7) is achieved. In Fig. 3 we plot this bound for a distortion measure that decays exponentially in rate: $D[P/\rho] = \exp[-0.1P/\rho]$. We plot the logarithm of the bound versus the reciprocal of the utilization rate to get a straight line that demarcates the achievable region. The shaded region below that line cannot be achieved by any algorithm. The queue is more busy (more arrivals per transmit opportunity) when ρ^{-1} is small, and less busy when ρ^{-1} is large.

3 Algorithm Design

Because of its finite size, the system memory can hold a small number of long, precise signal descriptions or a larger number of short, lossy descriptions. In the former case the queue overflows more frequently than in the latter. Intuitively, then, we must balance the distortion caused by queue overflow against the distortion caused by using short descriptions.

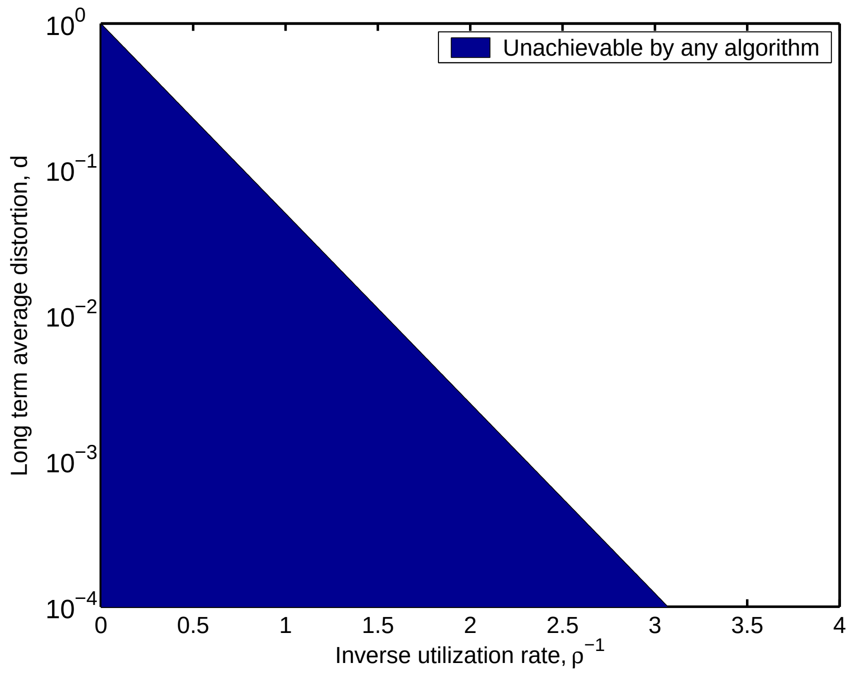

Figure 3: Bound on long term average distortion $d \geq D[P/\rho] = \exp[-0.1P/\rho]$ where $P = 30$.

In this section we present two algorithms that strike such a balance. The first picks a fixed description length for each signal. In effect, it treats all signals as distortion intolerant. The second uses multiresolution source codes so that it can adjust—in particular, shorten—signal description length dynamically to match the evolving state of the queue.

3.1 Baseline Algorithm

Consider the following baseline algorithm. If $\mathbf{s}_i$ arrives and there is space in the buffer, the queue stores it at some predetermined precision Q (measured in bits), incurring quantization distortion $D[Q]$. If, as depicted in Fig. 4-a, the buffer is full when the signal arrives then the signal cannot be stored and is lost. Signals are transmitted one at a time on a first-in first-out (FIFO) basis, as shown in Fig. 4-b.

Baseline Algorithm

Initialization: Divide the queue memory M into M/Q blocks of size Q.

Storage: If the queue is not full, assign an arriving signal to an available memory block, encoding the signal at distortion $D[Q]$. If the queue is full, the signal is lost, incurring distortion $d_{\max} = D[0]$.

Transmission: Send bits from a single encoded signal until the whole description is sent, then switch to

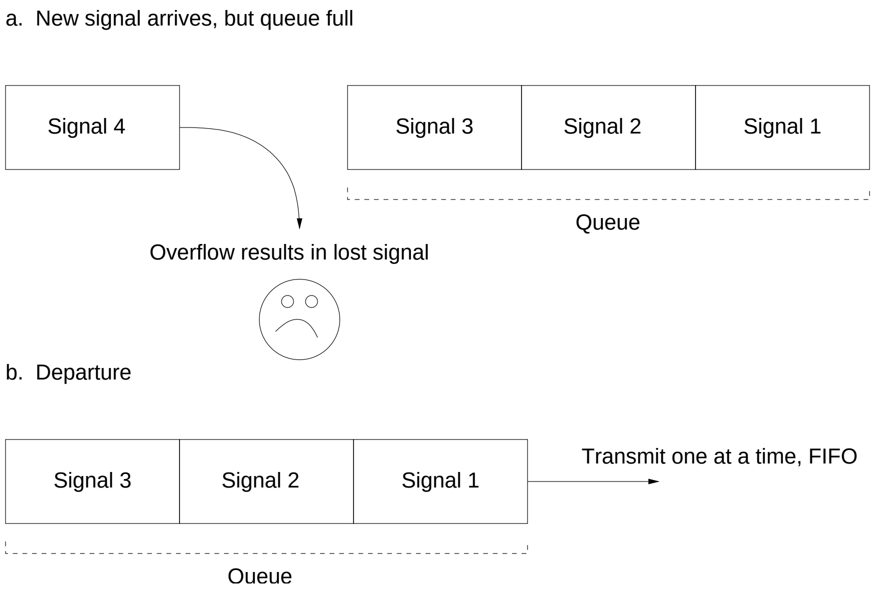
a. New signal arrives, but queue full
Signal 4
Signal 3
Signal 2
Signal 1
Queue
Overflow results in lost signal
b. Departure
Signal 3
Signal 2
Signal 1
Transmit one at a time, FIFO
Oueue

Figure 4: Basic idea of baseline algorithm. Signals are assigned static memory blocks and are transmitted one at a time. If a signal arrives when the queue is full, it is lost.

the next encoded signal.

The queue memory M should be an integer multiple of Q, allowing the storage of $K = M/Q$ signals, but as mentioned in Section 2 we relax integer constraints in this paper. As long as $M \gg Q$ this relaxation does not greatly affect the results.

Of the N_{arr} signals that arrive in $[0, T]$, some number are handled by the queue while the rest overflow and are lost. Let $N_{\text{lost}} \leq N_{\text{arr}}$ denote the number of the latter. The average distortion d_T can be bounded as

$$0 \leq d_T - \left(\left(1 - \frac{N_{\text{lost}}}{N_{\text{arr}}}\right) D[Q] + \frac{N_{\text{lost}}}{N_{\text{arr}}} d_{\max} \right) \leq \frac{K}{N_{\text{arr}}} (d_{\max} - D[Q]). \tag{8}$$

The gap between the upper and lower bounds arises from signals that arrive before T but may still be enqueued at time T. As N_{arr} grows with T the right side of (8) approaches zero and the bounds converge, yielding the approximation

$$d_T \simeq \left(1 - \frac{N_{\text{lost}}}{N_{\text{arr}}}\right) D[Q] + \frac{N_{\text{lost}}}{N_{\text{arr}}} d_{\max}. \tag{9}$$

A higher quantization rate leads to a smaller first term in (9), but it also increases the likelihood of overflows, thereby increasing the second term in (9). The designer's challenge is to balance these two mechanisms.

3.2 Multiresolution Source Codes

Multiresolution source codes are composed of ordered subcodes $\mathcal{C}_1, \mathcal{C}_2, \ldots$ of rates $R_1, R_2, \ldots$, respectively. Distortion $D[\sum_{i=1}^{k} R_i]$ is achieved for $k = 0, 1, 2, \ldots$ when the first k codewords are available to the decoder. For random sources, if a multiresolution source code is optimal at each step, i.e., if $R(D[\sum_{i=1}^{k} R_i]) = \sum_{i=1}^{k} R_i$, where $R(\cdot)$ is the rate-distortion function for the source-distortion pair under consideration, it is called a *successively refinable* source code [4, 13].

As a first example of a multiresolution source code, consider a scalar source s which is a random variable uniformly distributed on $[0, 1]$. Fidelity is measured by absolute distortion $\mathsf{d} = |\mathsf{s} - \hat{\mathsf{s}}|$. Using nested uniform scalar quantizers of rates $R_1, R_2, \ldots$, an expected distortion-rate trade-off $E[\mathsf{d}] = 2^{-2(1+\sum_{i=1}^{k} R_i)}$ can be achieved for $k = 0, 1, 2, \ldots$.

As a second example, consider consider a length-n vector source $\mathbf{s}$ consisting of independent identically distributed zero-mean Gaussian random variables of variance σ^2, and a squared distortion measure $\mathsf{d} = \frac{1}{n}\sum_{j=1}^{n}(\mathsf{s}_j - \hat{\mathsf{s}}_j)^2$. As n gets large, it is shown in [4] that this source-distortion pairing approaches successively refinability. Successive refinability for this pairing means that the average distortion $E[\mathsf{d}] = \sigma^2 2^{-2\sum_{i=1}^{k} R_i}$ is achievable for $k = 0, 1, 2, \ldots$.

3.3 Universal Algorithm: Distortion Control

Consider how multiresolution source codes allow us to modify the baseline algorithm. If a new signal arrives when the buffer is full, instead of losing that signal completely, we make room for it by deleting the least significant information in the queue. Figure 5-a indicates the mechanics of this “squeeze” algorithm. Conversely, when a packet is transmitted, the most significant information in the queue is transmitted first. This transmission protocol tends both to increase the likelihood that more significant information reaches its destination and to reduce the delay on that information.

These prioritized storage and extraction strategies are greedy since they only look one step into the future when determining memory allocations. As the number of arrivals and departures grows, determining what to keep in memory and what to transmit become more complex. In Section 3.3.1 and 3.3.2 we turn these greedy decision problems into tractable optimization problems. In Section 3.3.3 and Appendix A we show that the greedy perspective does not lead to a performance loss: no other algorithm can attain a lower d_T for any T.

For many applications it is necessary to introduce granularity in the size of the subcodes, e.g., packet can only be composed of an integer number of bits from each source. While the optimization problems of

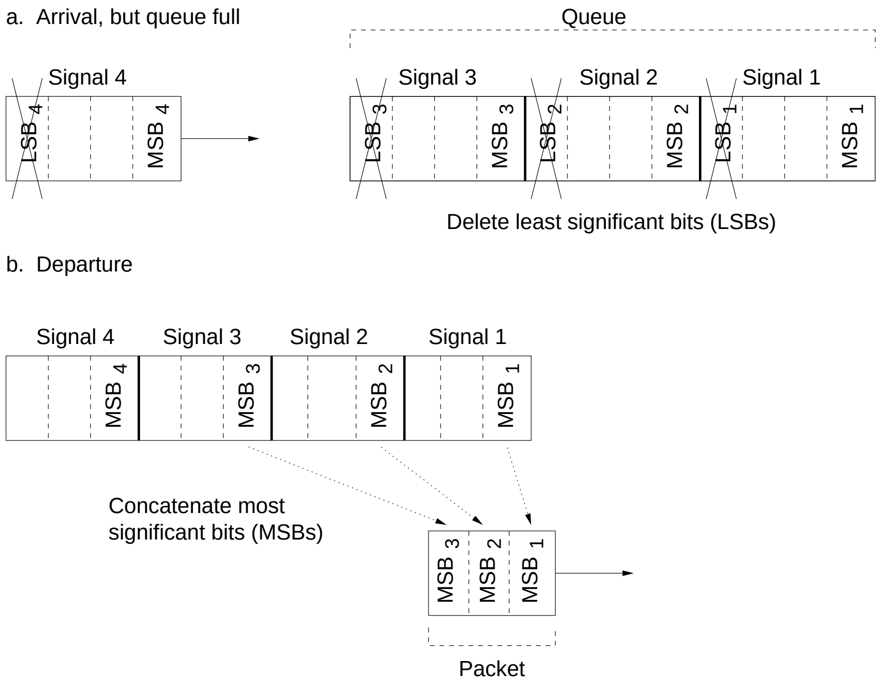

Figure 5: Basic idea of universal approach. Signals are stored in a progressive manner. Most significant information is transmitted first, and if a new signal arrives when the queue is full, all signals are "squeezed" to fit at lower fidelities.

Section 3.3.1 and 3.3.2 are presented without integer constraints, we impose such constraints explicitly when showing sample-path optimality in Section 3.3.3.[2]

3.3.1 Extraction Algorithm

Suppose signals $\mathbf{s}_1, \ldots, \mathbf{s}_{m-1}$ have arrived in the interval $[0, t]$ and the next event is a packet departure. Define B_i^- and Q_i^-, $i = 1, \ldots, m-1$, respectively as the number of bits describing $\mathbf{s}_i$ already at the decoder and still retained in the queue's memory at time t. Equivalent to minimizing average distortion, we minimize the cumulative distortion at the decoder, $\sum_{i=1}^{m-1} D[B_i^-]$. We determine the number of bits δ_i from each signal's encoding to include in the next packet to minimize the distortion after the packet is received. Again note that the δ_i are not constrained to be integers. The decoder distortion after this packet is received is $\sum_{i=1}^{m-1} D[B_i^+]$ where $B_i^+ = B_i^- + \delta_i$.

We use Lagrange multipliers to optimize the δ_i. The problem is constrained so that the number of transmitted bits does not exceed the size of the packet, i.e., $\sum_{i=1}^{m-1} \delta_i \leq P$, and so that $0 \leq \delta_i \leq Q_i^-$ for all i. Temporarily

[2]Sample-path optimality is also obtained in the absence of integer constraints, though the argument is less intuitive.

ignoring the latter constraints, the cost functional is

$$\mathcal{L} = \sum_{i=1}^{m-1} D[B_i^- + \delta_i] + \gamma \left(\sum_{i=1}^{m-1} \delta_i - P \right). \tag{10}$$

If the optimal δ_i are between zero and Q_i^- for all i, they can be found by differentiating (10) with respect to δ_i:

$$\frac{d\mathcal{L}}{d\delta_i} = D'[B_i^- + \delta_i] + \gamma = 0. \tag{11}$$

The convexity of $D[\cdot]$ implies that $D'[\cdot]$ is monotonic, hence (11) tells us that the best transmission policy attempts to even out the description rates at the decoders. This is in keeping with the intuition given by the bound (7) that an optimal, but usually unachievable, policy is to send all signals at a constant rate equal to the average throughput of the system.

Sometimes the policy of making the decoder description rates equal is not implementable because of constraints on packet size or because the δ_i must be non-negative. To optimize the δ_i while taking into account these active constraints we use the Kuhn-Tucker conditions.

Theorem 1 (Transmit Packet) *Let $D[\cdot]$ be convex, and let $\{Q_i^-\}$ and $\{B_i^-\}$ define the queue and decoder contents, respectively, prior to transmission. Then the number of bits $\{\delta_i\}$ to transmit to minimize the distortion $\sum_{i=1}^{m-1} D[B_i^- + \delta_i]$ is:*

$$\delta_i = \begin{cases} 0 & \text{if } \Delta_i < 0, \\ \Delta_i & \text{if } \Delta_i \in [0, Q_i^-], \\ Q_i^- & \text{if } \Delta_i > Q_i^-, \end{cases} \tag{12}$$

for $i = 1, \ldots, m-1$, where $\Delta_i = \gamma - B_i^-$ and where γ is chosen to fill the departing packet, i.e., $\sum_{i=1}^{m-1} \delta_i = P$. If fewer than P bits are enqueued, all are transmitted.

After transmission, the decoder holds $B_i^+ = B_i^- + \delta_i$ bits and the queue holds $Q_i^+ = Q_i^- - \delta_i$ bits for each signal $\mathbf{s}_i$.

This method for determining which bits to transmit is akin to "water-filling" for colored Gaussian channels in channel-coding theory, and is illustrated in Fig. 6.

3.3.2 Storage Algorithm

Again suppose that signals $\mathbf{s}_1, \ldots, \mathbf{s}_{m-1}$ have arrived in the interval $[0, t]$, but now the next event is the arrival of signal $\mathbf{s}_m$. The buffer contents at time t, Q_i^-, $i = 1, \ldots, m-1$, upper-bound the buffer contents Q_i^+ after $\mathbf{s}_m$ is stored. Since, at most, all M bits of memory can be assigned to $\mathbf{s}_m$, for convenience set $Q_m^- = M$. Define δ_i, $i = 1, 2, \ldots, m$, to be the number of bits discarded from each signal to store $\mathbf{s}_m$. We determine the $\{\delta_i\}$ that minimize the a posteriori cumulative distortion across the system, $\sum_{i=1}^{m} D[Q_i^- + B_i^- - \delta_i]$. We include bits still in the queue since they may be transmitted at some future time.

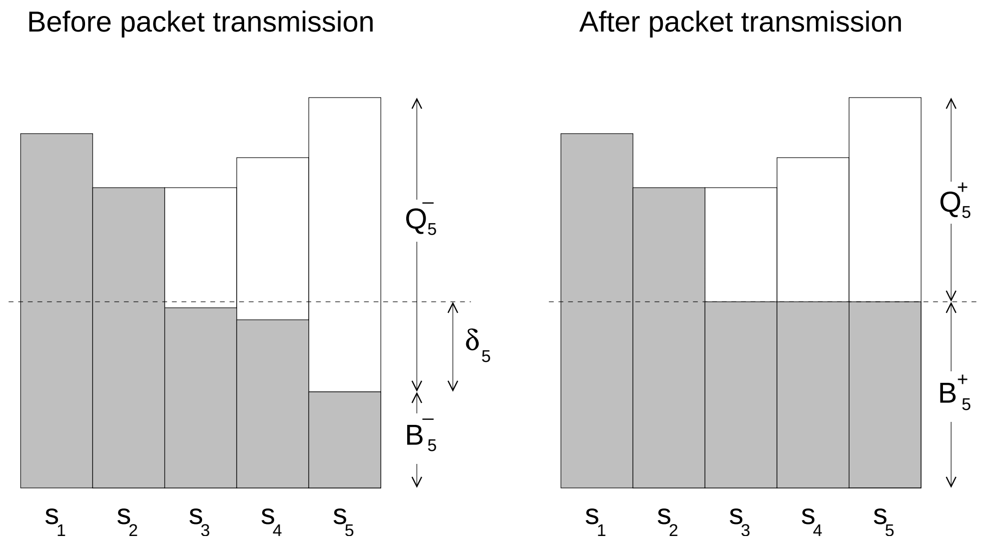
Before packet transmission
After packet transmission
Q_5^-
δ_5
B_5^-
Q_5^+
B_5^+
s_1 s_2 s_3 s_4 s_5
s_1 s_2 s_3 s_4 s_5

Figure 6: Determine packet contents by "water-filling" to the dashed line, thereby attempting to equalize signal description rates at the destinations.

The number of enqueued bits cannot exceed the queue memory, i.e., $\sum_{i=1}^{m}(Q_i^- - \delta_i) = \sum_{i=1}^{m} Q_i^+ \leq M$ where $0 \leq \delta_i \leq Q_i^-$ for all i. Making the queue memory constraint an equality, which minimizes distortion, and temporarily ignoring the other constraints yields the Lagrangian cost functional

$$\mathcal{L} = \sum_{i=1}^{m} D[B_i^- + Q_i^- - \delta_i] + \gamma\left(\sum_{i=1}^{m}(Q_i^- - \delta_i) - M\right). \tag{13}$$

If the optimal δ_i are strictly between zero and Q_i^- for all i, they can be determined by differentiating (13) with respect to δ_i:

$$\frac{d\mathcal{L}}{d\delta_i} = D'[B_i^- + Q_i^- - \delta_i] - \gamma = 0. \tag{14}$$

Equation (14) indicates that when storing a new signal, the best policy attempts to even out the overall signal description rates (in the queue and at the decoder).

Sometimes it is not possible to even out the description rates because of the constraints on the δ_i. To optimize the δ_i while taking into account these active constraints we use the Kuhn-Tucker conditions. This method for determining which bits to discard is illustrated in Fig. 7 and results in the following theorem:

Theorem 2 (Store Signal) *Let $D[\cdot]$ be convex and let $\{Q_i^-\}$ and $\{B_i^-\}$ define the queue and decoder contents, respectively, prior to the storage of signal $\mathbf{s}_m$, where $Q_m^- = M$ and $B_m^- = 0$. Then the number of bits $\{\delta_i\}$ to discard to minimize the system distortion $\sum_{i=1}^{m} D[Q_i^- + B_i^- - \delta_i]$ is:*

$$\delta_i = \begin{cases} 0 & \text{if } \Delta_i < 0, \\ \Delta_i & \text{if } \Delta_i \in [0, Q_i^-], \\ Q_i^- & \text{if } \Delta_i > Q_i^-, \end{cases} \tag{15}$$

where $\Delta_i = B_i^- + Q_i^- + \gamma$ and γ is chosen to keep the queue memory full, i.e., $\sum_{i=1}^{m}(Q_i^- - \delta_i) = M$. If the queue has memory available to store $\mathbf{s}_m$ losslessly, no bits are discarded.

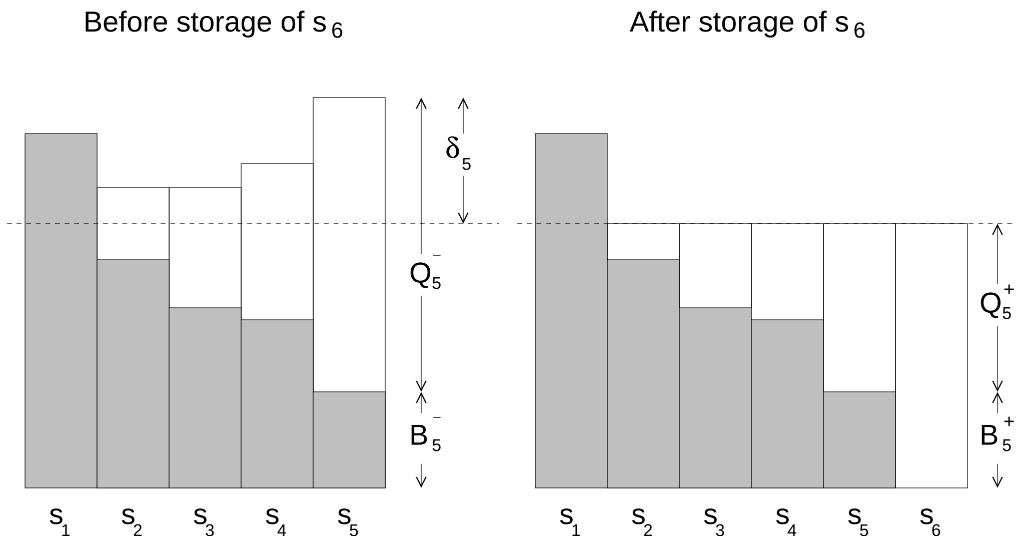

Figure 7: Determine memory allocations by "water-filling" to the dashed line, thereby attempting to equalize system-wide signal description rates.

After storage, the decoder holds $B_i^+ = B_i^-$ bits and the queue holds $Q_i^+ = Q_i^- - \delta_i$ bits for each signal $\mathbf{s}_i$.

Universal Algorithm with Distortion Control

Transmission:

(a) Calculate $\{\delta_i\}$ according to Thm. 1.

(b) Concatenate the most significant δ_i bits of each $\mathbf{s}_i$ into a packet.

(c) Transmit the packet.

(d) Calculate $B_i^+ = B_i^- + \delta_i$ and $Q_i^+ = Q_i^- - \delta_i$.

Storage:

(a) Calculate $\{\delta_i\}$ according to Thm. 2.

(b) Discard the δ_i least significant stored bits of each signal $\mathbf{s}_i$.

(c) Store $\mathbf{s}_m$ with a multiresolution encoding at rate $Q_m^+ = M - \delta_i$.

3.3.3 Sample-Path Optimality of Universal Algorithm

A queuing algorithm is said to be sample-path optimal if for all realizations of arrivals and departures on the interval $[0, T]$, there exists no other algorithm with a lower average decoder distortion d_T at time T.

In Appendix A we prove that the universal algorithm introduced in the previous section is sample-path optimal. More generally, all "greedy" algorithms are sample-path optimal. This implies a stronger result: the universal algorithm simultaneously achieves the minimum possible distortion d_t at every time $t \in [0, T]$.

In contrast to the continuous development in the rest of the paper the proof in Appendix A uses a discrete argument. While the result can be shown in either a discrete or continuous context, the discrete version is more intuitive, and the transformation that lies at the heart of the proof may be useful in proving the optimality of other greedy algorithms.

4 Analysis and Comparison of Algorithms

In this section we compare the long term average performance of the baseline and universal algorithms for a simple queuing process. The arrival stream of signals is modeled as a Poisson process of rate λ. In an interval of τ seconds $\lambda\tau$ signals are expected to arrive. At the output of the queue, packets of P bits are emitted according to Poisson process of rate μ that is independent of the arrival process. This gives an average transmission rate of $P\mu$ bits per second. If, for instance, the quantization rate Q of the baseline algorithm equals the packet size P then the processes form a standard $M/M/1$ queue.

4.1 Steady State Performance of Baseline Algorithm

In this section we discuss the steady state performance of the baseline algorithm. These results provide a benchmark against which we compare the long term performance of the universal algorithm. In Section 3 we developed an approximation (9) on the average distortion d_T achieved by the baseline algorithm that becomes exact as T gets large:

$$d_T \simeq \left(1 - \frac{N_{\text{lost}}}{N_{\text{arr}}}\right) D[Q] + \frac{N_{\text{lost}}}{N_{\text{arr}}} d_{\max}. \tag{16}$$

Under the queuing model the right side of (16) converges as T grows. Therefore, letting $d = \lim_{T\to\infty} d_T$, we get the following expression for steady state average distortion:

$$\begin{aligned} d = \lim_{T\to\infty} d_T &= \left(1 - \lim_{T\to\infty} \frac{N_{\text{lost}}}{N_{\text{arr}}}\right) D[Q] + \lim_{T\to\infty} \frac{N_{\text{lost}}}{N_{\text{arr}}} d_{\max} \\ &= (1 - \Pr[\text{signal loss}])D[Q] + \Pr[\text{signal loss}] d_{\max}. \end{aligned} \tag{17}$$

A signal is "lost" when it arrives to find the queue full and so it cannot be stored. Because the Poisson processes is memoryless the steady state probability of signal loss equals equals the steady state probability that the queue is full.

If the utilization rate $\rho = \lambda/\mu$ is known the designer can optimize the precision Q at which signals are stored. We term this the 'optimized' baseline algorithm. If ρ is not known the universal algorithm becomes particularly attractive because, by the sample-path results of Section 3.3.3, its performance is at least as good as that of the optimized baseline algorithm. any choice of Q.

4.2 Comparison of Algorithms

In this section we present simulation results for both algorithms. We focus on exponential distortion-rate trade-offs, $D[B_i] = \exp(-\alpha B_i)$.

In Fig. 8 we plot the long term average distortion performances of the algorithms versus ρ^{-1}, the inverse of the utilization rate, for $M = 1200$, $P = 30$, and $\alpha = 0.1$. The experimental performance of the baseline algorithm when we select the maximum number of signals the buffer can handle at a single time to be $K = 120$, $K = 40$, and $K = 20$, is plotted with the dotted curves. Experimental points are indicated by $\bullet$s, $+$s, and $\times$s, respectively. The first is a low-rate situation where three signal descriptions fit in a single packet (i.e., $\frac{P}{M/K} = 3$). The second is a medium-rate situation where each packet describes a different signal. The third is a high-rate situation where it takes two packets to describe each signal. The dashed curve plots the performance of the numerically-optimized baseline algorithm; each data point is indicated by a $*$. The solid curve is the performance bound of Fig. 3 given by (7). The experimental performance of the universal algorithm, indicated by $\circ$s, closely approximates the bound.

Figure 8 shows that in some regimes the performance of the optimized baseline algorithm can be quite close to the lower bound, in particular when ρ^{-1} is small (i.e., the queue is busy). If ρ is known in such situations, the baseline algorithm can be used with little loss of performance vis-a-vis the bound (7), allowing us to capitalize on the computational simplicity of that algorithm. However, the larger ρ^{-1} is (the less busy the queue is), the larger the performance gap between universal and baseline algorithms. A further disadvantage of the baseline algorithm is also illustrated by the figure: the performance of the baseline algorithms depends markedly on knowledge of ρ. In situations where ρ is uncertain, the baseline algorithm cannot guarantee good results. In these cases the sample-path optimality of the universal algorithm becomes more attractive.

Figure 8 also illustrates the changing character of a given baseline design as a function of utilization rate. Any baseline design has two distinct regions of operation, separated by the break point where its performance comes closest to the bound.

In the "communication-constrained" region ρ^{-1} is smaller (the queue is more busy) than the given baseline algorithm is designed for. In this region the distortion is dominated by buffer overflow. As the queue gets busier, the probability of buffer overflow increases, and performance worsens. To lower distortion in

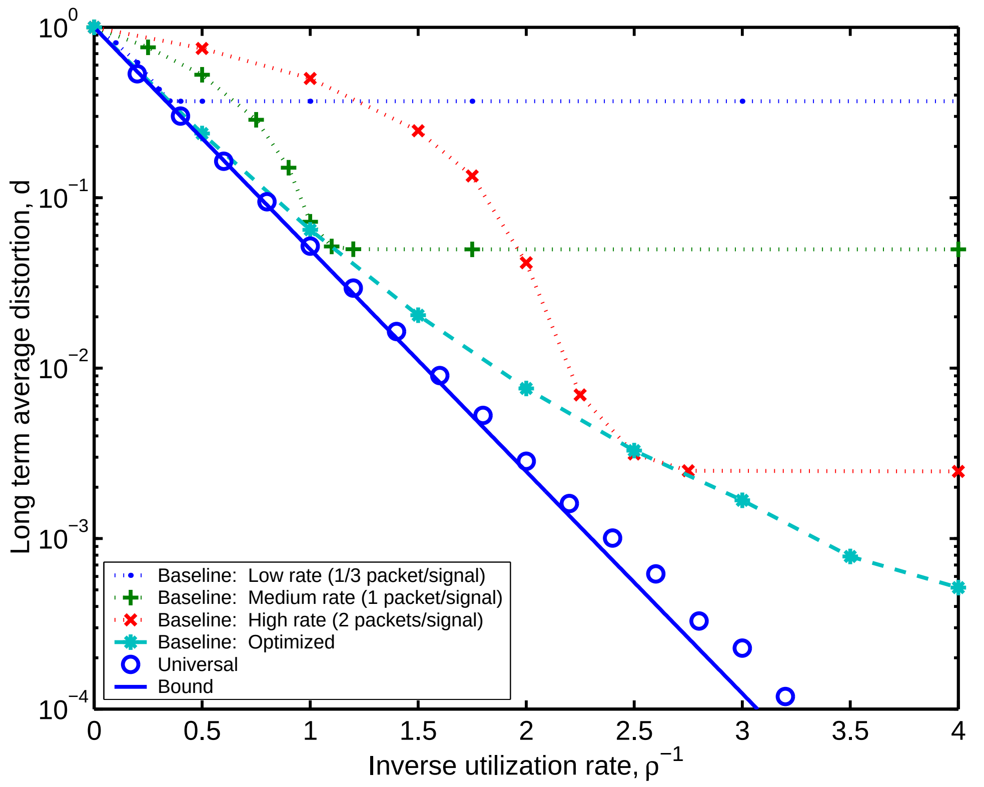

Figure 8: Long term average distortion versus inverse utilization rate for various-rate baseline algorithms, universal algorithm, and bound.

this region the probability of overflow must be reduced. This can be accomplished by decreasing signal quantization rate, which means that it takes less time to transmit each signal in the queue. The beneficial effect of decreasing the quantization rate in the communication-constrained region can be seen by noting that the lowest-rate algorithm performs best in the most-busy region near $\rho^{-1} = 0$.

In the “quantization-constrained” region ρ^{-1} is bigger (the queue is less busy) than the given baseline algorithm is designed for. In this region the distortion is dominated by quantization noise, which is invariant to changes in utilization rate. To lower distortion in this region the quantization rate must be increased. The beneficial effect of increasing quantization rate in the quantization-constrained region can be seen by noticing that the highest-rate algorithm performs best in the least-busy region near $\rho^{-1} = 4$.

The results of this section suggest a possible hybrid algorithm. Instead of using hierarchical source codes, consider an algorithm that varies the fidelity at which new signals are stored depending on the observed history of queue overflows. If most signals are lost due to overflows, the quantization rate should be reduced. On the other hand, if the memory is free most of the time, the quantization rate should be increased. The quantization rate should be controlled as a function of the empirical probability of buffer overflow to stay close to the performance of the optimized baseline algorithms strikes a good balance between overflow and quantization distortion.

5 Concluding Comment

In this paper we have focused on signals that share a common distortion measure. The universal algorithm can be applied to more general problems where the distortion measure is signal specific. This would be particularly relevant for systems that handle different classes of data, e.g., audio versus image or video. Different distortion measures could also be used to give different qualities-of-service to different data classes, or data sources, and to implement a fair allocation of resources between those data classes or sources. Distortion measures can be designed, for example, to give priority to delay-sensitive data (such as voice), to favored data (as in the digital camera example), and even to be time-dependent so that the marginal utility of each piece of information decreases the longer it remains enqueued. As long as all distortion measures are convex, the priority storage and transmission protocols presented herein will continue to be sample-path optimal.

A Sample-path Optimality of Greedy Queuing with Distortion Control

In this appendix we show that the universal algorithm described in Sec. 3.3 is sample-path optimal. The proof is developed by establishing two intermediate results:

I. All greedy protocols (defined below) achieve the same distortion.

II. Any optimal sample path can be transformed into a greedy sample path with the same—optimal—distortion. Thus there exists a greedy optimal path.

Combining I and II proves that all greedy protocols are sample-path optimal.

We simplify the proof by taking the smallest quanta of signal description and transmission to be 1 bit, so that sets are discrete and finite. Any other quantization step size serves equally well; one can develop a proof for the continuous-valued algorithm described in Sec. 3.3 by taking the limit of decreasing step sizes.

A.1 Definitions

A *receive/transmit sample path* is a sequence of receive events and transmit opportunities that occur during a finite time interval $[0, T]$. Each receive event $i = 1, \ldots, N$ or transmit opportunity $i' = 1, \ldots, N'$ occurs at some time t_i or t'_i, respectively. Transmit opportunities are described by the maximum number $l_{i'}$ of bits that may be transmitted. Receive events are described by the full-length encoding S_i of the received signal, limited in length to the queue storage size M, and by the distortion function $D[\cdot]$ of the signal[3]. We view a receive/transmit sample path as being determined by nature, out of control of the system designer.

A *queue sample path* is the result of applying a queuing discipline to a receive/transmit sample path. In a queue sample path each transmit opportunity is further described by the set of bits transmitted during that opportunity. A new type of event is also included: drop events are described by the time t_i of the event and the set of bits that are discarded. We will see presently that, to minimize distortion, drop events should immediately follow receive events, so that they may be identified using the same set of event indices i.

A queue sample path must satisfy the obvious state evolution rules: receive events add to the queue, transmit and drop events subtract from the queue, and drop events must be used to keep the queue size at or below its maximum.

The distortion at the receiver is determined by the distortion-rate function of each signal and by the bits

[3]The proof may be immediately generalized by using a different distortion function for each signal.

delivered to the receiver thus far. Let B_i be the set of bits available at the receiver for signal S_i. The total distortion d_T at the receiver is the sum

$$d_T = \sum_{i=1,\ldots,N} D[B_i] \tag{18}$$

of the distortions of the individual signals. We require the distortion function $D[\cdot]$ to be expressible as a sum of distortion contributions of each bit:

$$D[B_i] = D[0] - \sum_{b \in B_i} \tilde{D}(b), \tag{19}$$

where $D[0]$ is a constant and $\tilde{D}(b) \geq 0$ is the *marginal utility* of bit b. For convenience, we number the bits of S_i sequentially and assume they are arranged in order of decreasing marginal utility, i.e.,

$$\tilde{D}_i(j) \geq \tilde{D}_i(j+1) \tag{20}$$

for $j = 1, 2, \ldots$.

For many source encodings, such as the multiresolution source codes described in Sec. 3.2, the jth bit of S is worthless unless the initial bits $1, \ldots, j-1$ are also received. We refer to this requirement as the *sequential constraint*. The sequential constraint violates the additive model (19), but—so long as the ordering constraint (20) is maintained—this turns out not to affect our results. We will show that there is no advantage to transmitting bits of lesser marginal utility in advance of those of greater marginal utility. Note that the sequential constraint combined with the ordering constraint (20) is a discrete analog to a convexity constraint on the distortion function $D[\cdot]$, if $D[\cdot]$ is viewed as a function of the number of descriptive bits. We emphasize that convexity is an essential requirement for our results in applications where the sequential constraint prevails.

A queue sample path is *optimal* for a given transmit/receive sample path if there exists no other queue sample path with a lower distortion d_T at time T. This definition of optimal is rather strong, as the queuing discipline needed to arrive at an optimal path could be noncausal. Because the queue, events, and time horizon are finite or discrete or both, there always exists at least one optimal path, and in general there will be many.

A queue sample path is *greedy* if it meets five conditions, defined more precisely in the lemmas below: (a) drop events immediately follow receive events, (b) drop events discard the minimum number of bits necessary, (c) transmit opportunities are filled with the maximum number of bits possible, (d) transmit opportunities are serviced using enqueued bits of maximum marginal utility, and (e) drop events discard enqueued bits of minimum marginal utility.

It is easy to see that the universal algorithm produces greedy sample paths. Moreover, we immediately have result I above: every greedy queue sample path for a given receive/transmit sample path has the same distortion. This follows because greedy paths may differ only in how they break ties in conditions (d) and (e) above, yet these choices do not change the distortion. In other words, the bits used to describe signals are only distinguishable—in the sense of distortion minimization—based on their marginal utilities, i.e., on the amount a particular descriptive bit decreases the total distortion.

A.2 Proof

To prove our central result—that the "universal algorithm" is sample-path optimal—we use several variations of a common argument to establish result II above. Specifically, we introduce a local transformation of a queue sample path that either decreases distortion or leaves it unchanged. Applying this transformation repeatedly to an optimal path reaches a fixed point with some desired property, establishing that there exists an optimal path with the same property.

Lemma 1 *There exists an optimal path for which all drop events immediately follow receive events.*

Proof: Consider a queue sample path for which some drop event occurs at a time t strictly between the times t_i and t_{i+1} of two receive events i and $i+1$. Construct a new queue sample path in which the drop event instead occurs at time $t = t_i$. It is easy to see that this new path is feasible: queue occupancy is not increased, and all discarded bits are enqueued at the time of the drop event. Distortion is unchanged.

There are a finite number of drop events, hence repeated application of this transformation to an optimal queue sample path creates an optimal path in which all drop events immediately follow receive events. ■

Lemma 2 *There exists an optimal path for which all drop events discard the minimum number of bits necessary to meet the queue memory constraint.*

Proof: Consider a queue sample path for which there exists a drop event i that drives the queue occupancy below its maximum size M. Construct a new queue sample path in which the drop event i discards a subset of the bits previously discarded, the subset size selected to exactly meet the queue memory constraint. If the queue was not full prior to the drop event, discard no bits. Let $\mathcal{X}$ be the set of bits previously dropped but now retained in the new queue sample path.

The new queue sample path may be infeasible, as the queue occupancy may exceed M at various times subsequent to the modified drop event. To correct this, consider the next receive event $i+1$, together with the (optional) simultaneous drop event. If the queue occupancy exceeds M, modify the drop event $i+1$ to discard a sufficient number of bits in $\mathcal{X}$ to meet the memory constraint. This is always possible. Repeat for all subsequent receive events $i+2, i+3, \ldots$, depleting the set $\mathcal{X}$ as necessary. The final queue sample path is feasible and has the same distortion as the original.

Repeated application of this transformation to an optimal queue sample path creates an optimal path in which all drop events have the minimal size necessary to meet the memory constraint. ■

Lemma 3 *There exists an optimal path for which every transmit opportunity is maximally exploited. That is, the number of bits transmitted at opportunity i' equals either the number of bits in the queue at time $t_{i'}$ or the number of bits $l_{i'}$ available in the transmit opportunity, whichever is smaller.*

Proof: Consider a queue sample path for which there exists an underutilized transmit opportunity. Construct a new queue sample path in which the transmit opportunity is fully utilized by arbitrarily selecting additional bits from the queue for transmission. Delete these bits from subsequent drop or transmit events. The new queue sample path remains feasible, and its distortion is equal to or smaller than the original.

Repeated application of this transformation to an optimal queue sample path creates an optimal path in which all transmit opportunities are maximally exploited. ■

Lemma 4 *There exists an optimal path for which (i) every transmit opportunity is serviced using the enqueued bits of maximum marginal utility, and (ii) every drop event discards enqueued bits of minimum marginal utility.*

Proof: We prove the two statements of the lemma in parallel. Without loss of generality we view a transmit (resp. drop) event of size l as a sequence of l one-bit transmit (resp. drop) events.

Consider a queue sample path in which a bit b from signal S_i is transmitted (resp. dropped) time t. The marginal utility of b is $\tilde{D}(b)$. Let $\tilde{b}$ be a bit of greatest (resp. least) marginal utility over all enqueued bits at time t. If b and $\tilde{b}$ have the same marginal utility then do nothing. Otherwise, looking in to the future up to time T, the bit $\tilde{b}$ may be transmitted, dropped, or left in the queue. Construct a new queue sample path in which $\tilde{b}$ is transmitted (resp. discarded) at time t and b meets the fate previously assigned to $\tilde{b}$. It is easy to see that this transformation cannot increase distortion, for if both bits are transmitted (resp. dropped or held) the distortion does not change, while if $\tilde{b}$ was dropped or held (resp. transmitted) distortion decreases.

Repeated application of this transformation to an optimal queue sample path, working sequentially from time 0 to T, creates an optimal path in which all transmit opportunities are serviced using enqueued bits of maximum marginal utility and all drop events discard enqueued bits of least marginal utility. ■

Combining Lemmas 1–4 proves that there exists a greedy optimal path, thereby establishing result II and completing the proof.

References

[1] M. Alasti, K. Sayrafian-Pour, A. Ephremides, and N. Farvardin. Multiple description coding in networks with congestion problem. *IEEE Trans. Inform. Theory*, 47:891–902, March 2001.

[2] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *IEEE Trans. Inform. Theory*, 42:1737–1744, November 1996.

[3] J. G. Apostolopoulos, T. Wong, W. Tan, and S. J. Wee. On multiple descriptions streaming with content delivery networks. In *Proc. IEEE INFOCOMM*, June 2002.

[4] W. H. Equitz and T. M. Cover. Successive refinement of information. *IEEE Trans. Inform. Theory*, 37:269–275, March 1991.

[5] N. Farvardin and J. W. Modestino. Adaptive buffer-instrumented entropy-coded quantizer performance for memoryless sources. *IEEE Trans. Inform. Theory*, 32:9–22, January 1986.

[6] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Trans. Networking*, 7:458–472, August 1999.

[7] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidence. *IEEE/ACM Trans. Networking*, 1:397–413, August 1993.

[8] D. Harrison and J. W. Modestino. Analysis and further results on adaptive entropy-coded quantization. *IEEE Trans. Inform. Theory*, 36:1069–1088, September 1990.

[9] C. V. Hollot, V. Misra, D. Towsley, and W. Gong. Analysis and design of controllers for AQM routers supporting TCP flows. *IEEE Trans. Automat. Contr.*, 47:945–959, 2002.

[10] B. Prabhakar, E. Uysal-Biyikoglu, and A. El Gamal. Energy-efficient transmission over a wireless link via lazy packet scheduling. In *Proc. IEEE INFOCOMM*, pages 386–394, April 2001.

[11] R. Puri, K.-W. Lee, K. Ramchandran, and V. Bharghavan. An integrated source transcoding and congestion control paradigm for video streaming in the Internet. *IEEE Trans. Multimedia*, 3:18–32, March 2001.

[12] K. Ramchandran and M. Vetterli. Multiresolution joint source-channel coding. In H. V. Poor and G. W. Wornell, editors, *Wireless Communications: Signal Processing Perspectives*, chapter 7, pages 282–329. 1998.

[13] B. Rimoldi. Successive refinement of information: Characterization of the achievable rates. *IEEE Trans. Inform. Theory*, 40:253–259, January 1994.

[14] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Signal Processing*, 41:3445–3462, December 1993.

[15] D. Tse, R. Gallager, and J. Tsitsiklis. Optimal buffer control for variable-rate lossy compression. In *Proc. 31st Allerton Conf. on Communication, Control and Computing*, September 1993.

[16] D. N. C. Tse. *Variable-rate Lossy Compression and its Effects on Communication Networks*. PhD thesis, Mass. Instit. of Tech., 1994.