A Universal Approach to Queuing With Distortion Control

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

***Abstract*—An efficient buffer-management algorithm is developed for queues that handle distortion-tolerant data under finite memory limitations. We avoid overflows and realize significant performance gains through the use of multiresolution source codes. These codes enable us to reduce the fidelity of signal descriptions in a controlled progressive 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 noncausal knowledge of queue arrival and service times.**

***Index Terms*—Buffer management, congestion control, multimedia communications, multiresolution source coding, queuing analysis, successive refinement, transcoding.**

I. 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, the queue can use this characteristic to adjust fidelity as new signals arrive. In this note, 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 [2], [7], and the references therein) to exploit the inherent distortion-tolerance of signals. These codes prioritize source information from most significant to least significant. Such structure 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 should be deleted to free memory space. This leaves more significant information undisturbed. Conversely, most significant information should be transmitted first. This guarantees that such information is not lost in future overflows, and allows the source to be reconstructed progressively as the code stream becomes available at the queue output.

We formalize these intuitive ideas and show they form the basis for an optimal approach to buffer management. Regardless of arrival and departure times, no algorithm can attain a lower average distortion than the one proposed. The approach is universal and sample-path optimal: It operates without knowledge of the statistics of the arrival and departure processes, and its performance cannot be bettered by any algorithm over any set of arrival and departure times.

This buffer-management policy is a form of distortion-measure-dependent priority queuing. Whenever bits enter the system, they are assigned to queues of differing priorities according to their individual utilities. Bits in the highest-priority queue are transmitted first, and bits in the lowest-priority queue are dropped first. If a number of bits constitute one "signal," then there is often an explicit ordering to the bits—some bits are useless without knowledge of others. In these cases we assume that the bits are arranged in order of decreasing utility, which is analogous to the utility function being convex. Further, we assume an observation-type model of arrivals, whereby all the bits that constitute each signal arrive at the queue concurrently.

Consider image storage in a digital camera as one application of these ideas. The first pictures taken can be stored in memory (the buffer) at high resolution. As more pictures are taken, the resolution of all images can be progressively lowered in parallel to accommodate new data. The algorithms we develop provide a structured way to manage this resolution versus number-of-images trade off. We can also prioritize pictures so that favored pictures experience slower (or even zero) lowering of resolution. Such a system could prove useful in an autonomous sensor vehicle, such as a submarine, an interplanetary probe, or a crop monitor. System unpredictability may 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 achievable communication rates may vary as a result of changing environmental conditions.

Our setting is most closely related to that considered in a family of work on quantization for queuing (see, e.g., [3], [4], and [8]). In these papers, a variable-length source coder feeds a finite-memory queue that transmits the buffered information over a fixed-rate channel. The quantization rate is controlled based on the state of the queue to avoid overflows and to minimize average distortion. Our work differs because, by using multiresolution source codes, we effectively can change the quantization rate long after the source is quantized by deleting least significant information. Our ideas are also somewhat related to joint source-channel coding and multiple description coding for networks with packet losses (see, e.g., [1] and [5]). However, because we focus on a single-queue system, and assume a reliable output link, the application of our results to such networks is not direct. On the other hand, in our focused context, we are able to make much stronger statements about the optimality of our scheme.

The note is organized as follows. In Section II, we prove the main result—the sample-path optimality of the proposed algorithm. Then, to develop a sense of when distortion control leads to performance gains, and how large those gains are, in Section III, we describe a simple queuing model and develop a bound on the performance of any algorithm. Section IV presents two memory management algorithms: the sample-path optimal algorithm and a baseline algorithm for comparison. Section V compares the algorithms, and Section VI concludes the note.

II. Sample-Path Optimality of Greedy Queuing With Distortion Control

The policy we analyze in this section transmits the enqueued bits of maximum marginal utility at each transmit opportunity, and drops the enqueued bits of least marginal utility whenever faced with an overflow. This policy is greedy—it makes decisions that minimize the immediately resulting distortion, without looking further ahead. We show that the following result holds for convex-like additive distortion measures:

Manuscript received August 20, 2003; revised August 10, 2004. Recommended by Associate Editor R. S. Srikant. This work was supported in part by the National Science Foundation under Grant CCR-0073520, by MARCO/DARPA under Contract 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 is with the Department of Electrical Engineering and Computer Science, University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: sdraper@eecs.berkeley.edu).

M. D. Trott is with Hewlett-Packard Laboratories, Palo Alto, CA 94304 USA (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 USA (e-mail: gww@mit.edu).

Digital Object Identifier 10.1109/TAC.2005.844911

for all realizations of arrival and departure times, there exists no algorithm that achieves a lower average decoder distortion at any time than the one proposed. We term this sample-path optimality.

The proof is developed by establishing two intermediate results.

I) All greedy protocols (defined later) 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 one bit, so that sets are discrete and finite. Any other quantization size serves equally well; one can develop a proof for continuous-valued applications by taking the limit of decreasing quantization size.

A. 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.[1] We view a receive/transmit sample path as being determined by nature, out of the 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 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 indexes 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 function of each signal and by the bits 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

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

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, i.e.,

$$D[B_i] = D[0] - \sum_{b \in B_i} \tilde{D}(b) \tag{1}$$

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 utility

$$\tilde{D}_i(j) \geq \tilde{D}_i(j+1), \qquad \text{for } j = 1, 2, \ldots. \tag{2}$$

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

For many source encodings, such as the multiresolution source codes described in Section IV-B, 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 (1), but—so long as the ordering constraint (2) is maintained—this turns out not to affect our results. We show that there is no advantage to transmitting bits of lesser marginal utility in advance of those of greater utility. The sequential constraint combined with the ordering constraint (2) is a discrete analog to a convexity constraint on the distortion function $D[\cdot]$. 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 that follow: 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 algorithm proposed at the beginning of the section produces greedy sample paths. Moreover, we immediately have result I): 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), 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—by their marginal utilities, i.e., on the amount a particular descriptive bit decreases the total distortion.

B. Proof

To prove our central result—that the algorithm proposed is sample-path optimal—we use several variations of a common bit-swapping argument to establish result II). 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 the 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 meet the queue memory constraint exactly. 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 sample path.

The new queue sample path may be infeasible, as 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 drop event $i+1$ to discard a number of bits in $\mathcal{X}$ sufficient 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 sample path is feasible and, since the transmit opportunities have not been modified, has a distortion equal to the original.

Repeated application of this transform 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 view a transmit (respectively, drop) event of size l as a sequence of l one-bit transmit (respectively, drop) events.

Consider a queue sample path in which a bit b from signal S_i is transmitted (respectively, dropped) at time t. The marginal utility of b is $\tilde{D}(b)$. Let $\tilde{b}$ be a bit of greatest (respectively, 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 into 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 (respectively, dropped or held) the distortion does not change, while if $\tilde{b}$ was dropped or held (respectively, 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.

III. System Model and Performance Bound

In order to develop a sense of the performance gains of distortion-controlled queuing, we now provide a simple system model. At a high level, 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. Two 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. This is particularly well matched to situations where arrivals are sensor observations that must be quantized before being stored. Second, each packet departs over a shared link. This allows information about different signals to be concatenated into a single packet. It is most appropriate for settings where there is a single destination for all data, as in the sensor vehicle example, or in a broadcast setting where all destinations can listen in on a common transmission.

In the interval $[0, T]$, there are $N_{\rm arr}$ signal arrivals where the ith signal to arrive is denoted S_i. Let B_i denote the number of bits describing S_i transmitted to its destination by time T. The distortion in S_i at time T is given by the distortion function $D[B_i]$. The maximum distortion, incurred when the decoder has no information about a signal, is $d_{\max} \equiv D[0]$. Equivalent to minimizing the total distortion d_T, we can minimize the average distortion $\bar{d}_T$ where

$$\bar{d}_T = \frac{1}{N_{\rm arr}} d_T = \frac{1}{N_{\rm arr}} \sum_{i=1}^{N_{\rm arr}} D[B_i].$$

Ideally, we minimize this function for all T.

We assume that $D[B_i]$ is convex and bounded (hence monotonic and continuous) and—to simplify derivations—differentiable.[2] In order to ease the presentation of key ideas, for the rest of the note we focus on a common distortion measure for all S_i, and treat bits as infinitely divisible, thereby avoiding integer constraints in the optimization problems.

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

$$\sum_{i=1}^{N_{\rm arr}} B_i \le P N_{\rm dep} \tag{3}$$

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

We now derive a lower bound on $\bar{d}_T$ that is independent of memory size

$$\bar{d}_T \ge D\left[\frac{1}{N_{\rm arr}} \sum_{i=1}^{N_{\rm arr}} B_i\right] \ge D\left[P \frac{N_{\rm dep}}{N_{\rm arr}}\right] \tag{4}$$

where the first inequality follows from Jensen's inequality, and the second from (3). Assuming that the limits exist, define $\lambda = \lim_{T\to\infty} N_{\rm arr}/T$ to be the average arrival rate, $\mu = \lim_{T\to\infty} N_{\rm dep}/T$ 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 $d = \lim_{T\to\infty} \bar{d}_T$ follows. Since (4) holds for all T

$$d \ge D\left[P \lim_{T\to\infty} \frac{N_{\rm dep}}{T} \lim_{T\to\infty} \frac{T}{N_{\rm arr}}\right] = D\left[\frac{P\mu}{\lambda}\right] = D\left[\frac{P}{\rho}\right]. \tag{5}$$

We can move the limit inside and take the limits individually because $D[\cdot]$ is continuous, and because both limits exist by assumption. This bound indicates that 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 is achieved.

[2]The results of the note could, equivalently, be developed in terms of the concave utility function $\boldsymbol{d}_{\max} - \boldsymbol{D}[\boldsymbol{B}_i]$. 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).

IV. Algorithm Design

In this section, we present two queue management algorithms. The first is a baseline algorithm which sets a fixed description length for each signal. The second, universal algorithm, uses the ideas of Section II, implemented through multiresolution source coding, to adjust (in particular, shorten) signal description lengths dynamically to match the evolving state of the queue.

A. Baseline Algorithm

The following algorithm is computationally simple to implement, but its performance is quite dependent on parameter settings. Say that S_i arrives and there is space in the buffer. The queue stores S_i at some predetermined precision of Q bits. If the buffer can store at most M bits, then it can store at most $K = \lfloor M/Q \rfloor$ signals. If buffer is full, S_i cannot be stored, and is lost. Signals are transmitted one at a time in a first-in–first-out (FIFO) manner.

Of the $N_{\rm arr}$ signals that arrive in $[0, T]$, and the $N_0 \leq K$ signals in the queue at time 0, some number are handled by the queue while the rest overflow and are lost. Let $N_{\rm lost} \leq N_{\rm arr}$ denote the number of the latter. The average distortion $\bar{d}_T$ can be bounded as

$$0 \leq \bar{d}_T - \left[\left(1 - \frac{N_{\rm lost}}{N_{\rm arr}+N_0}\right) D[Q] + \frac{N_{\rm lost}}{N_{\rm arr}+N_0} d_{\max}\right] \leq \frac{K}{N_{\rm arr}+N_0}(d_{\max} - D[Q]). \quad (6)$$

The gap between the upper and lower bounds arises from signals that arrive before T, but are assumed to have all their bits still enqueued at time T. As $N_{\rm arr}$ grows much larger than K with T, the right-hand side of (6) approaches zero and the bounds converge, yielding the approximation

$$\bar{d}_T \simeq \left(1 - \frac{N_{\rm lost}}{N_{\rm arr}}\right) D[Q] + \frac{N_{\rm lost}}{N_{\rm arr}} d_{\max}. \quad (7)$$

A higher quantization rate leads to a smaller first term in (7), but it also increases the likelihood of overflow, thereby increasing the second term in (7).

B. Multiresolution Source Codes

Multiresolution source codes are composed of ordered sub-codes $\mathcal{C}_1, \mathcal{C}_2, \ldots$ 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. 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 [2], [6].

C. Universal Algorithm: Distortion Control

While multiresolution source codes encode a source in a manner compatible with the distortion control policy presented in Section II, determining what to keep in memory and what to transmit as the number of arrivals and departures grow becomes complicated. We now show how to optimize these decisions. First, we show how to choose the contents of any packet transmission. Suppose signals $S_1, \ldots, S_{m-1}$ 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 S_i already at the decoder and still enqueued at time t. If δ_i bits from signal i are included in the next packet, the total distortion after the packet is transmitted is $\sum_{i=1}^{m-1} D[B_i^+]$ where $B_i^+ \equiv B_i^- + \delta_i$.

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 ignoring the latter constraints, the cost functional is

$$\mathcal{L} = \sum_{i=1}^{m-1} D[B_i^- + \delta_i] + \gamma \sum_{i=1}^{m-1} \delta_i.$$

If the optimal δ_i are between zero and Q_i^- for all i, they can be found via differentiation

$$\frac{d\mathcal{L}}{d\delta_i} = D'[B_i^- + \delta_i] + \gamma = 0. \quad (8)$$

The convexity of $D[\cdot]$ implies that $D'[\cdot]$ is monotonic, hence (8) tells us that the best policy is to even out the description rates at the decoders. Often, we cannot equalize the rates because of constraints on packet size or because the $\delta_i \geq 0$. To take into account these active constraints we use the Kuhn–Tucker conditions, giving:

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

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

for $i = 1, \ldots, m-1$, where $\Delta_i = \nu - B_i^-$ and where ν 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.

This method for determining which bits to transmit is akin to "water-filling" for colored Gaussian channels in channel-coding theory. This is illustrated in Fig. 1 where ν the water-filling level. We can interpret these results in terms of priority queuing by considering each level-line (for instance, the dashed line) to correspond to a different priority. The lower the level, the more significant the information, and the higher the priority.

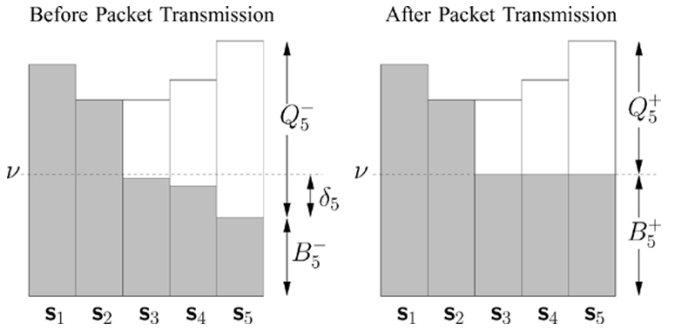

Fig. 1. Determine packet contents by "water-filling" to $\boldsymbol{\nu}$ (the dashed line). This attempts to equalize signal description rates at the destinations. After transmission, the decoder has $\boldsymbol{B_i^+ = B_i^- + \delta_i}$ bits and the queue $\boldsymbol{Q_i^+ = Q_i^- - \delta_i}$ bits.

A similar optimization dictates what to throw out when faced with an overflow. Say that the overflow event is the arrival of signal S_m. The Q_i^- upper-bound the buffer contents Q_i^+ after S_m is stored. Since, at most, all M bits of memory can be assigned to S_m, for convenience set $Q_m^- = M$. Define the δ_i to be the number of bits discarded from each signal to store 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 as they may yet be transmitted.

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 yields the Lagrangian

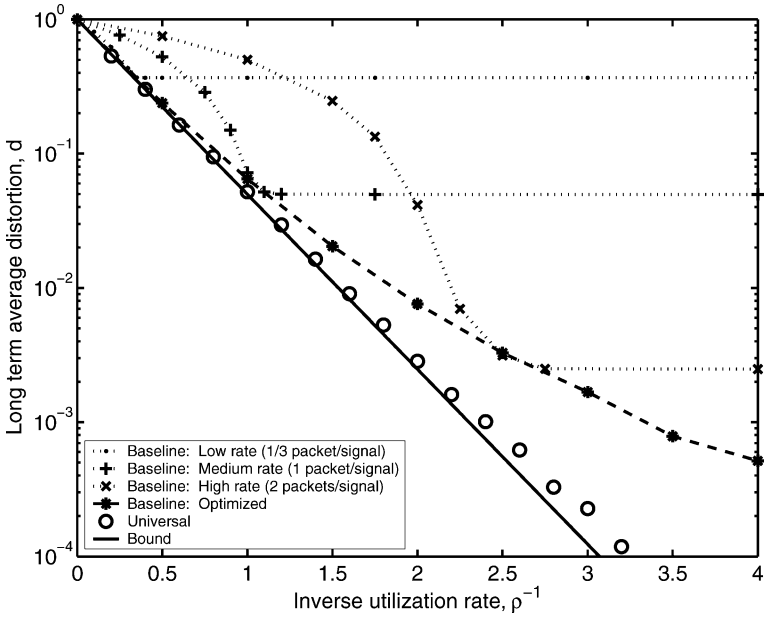

Fig. 2. Long term average distortion versus inverse utilization rate for various-rate baseline algorithms, universal algorithm, and bound.

cost functional $\mathcal{L} = \sum_{i=1}^{m} D[B_i^- + Q_i^- - \delta_i] - \gamma \sum_{i=1}^{m} (Q_i^- - \delta_i)$, giving the following.

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 S_m, where $Q_m^- = M$ and $B_m^- = 0$. The number of bits $\{\delta_i\}$ to discard to minimize system distortion $\sum_{i=1}^{m} D[Q_i^- + B_i^- - \delta_i]$ is

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

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

V. 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 arriving stream is a Poisson process of rate λ. 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. 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.

A. Steady-State Performance of Baseline Algorithm

The steady-state performance of the baseline algorithm provides a benchmark against which we compare the performance of the universal algorithm. The approximation (7) to the average baseline distortion $\bar{d}_T$ becomes exact as T gets large. Letting $d = \lim_{T\to\infty} \bar{d}_T$, we get

$$d = \lim_{T\to\infty} \bar{d}_T = (1 - \Pr[\text{signal loss}])D[Q] + \Pr[\text{signal loss}]d_{\max}$$

where $\Pr[\text{signal loss}] = \lim_{T\to\infty} N_{\text{lost}}/N_{\text{arr}}$. Because the Poisson processes is memoryless the steady-state probability of signal loss 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.

B. Comparison of Algorithms

We now present simulation results for both algorithms. We use an exponential distortion function, $D[B_i] = \exp(-0.1B_i)$. In Fig. 2, we plot the long term average distortion performances of the algorithms versus ρ^{-1}, for total memory $M = 1200$ and packet size $P = 30$. The experimental performance of the baseline algorithm for three different quantization rates 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 (three signals per packet). The second is a medium-rate situation (one signal per packet), and the third is a high-rate situation (two packets per signal). The dashed curve plots the performance of the optimized baseline algorithm; data points are indicated by $*$s. The solid line is the performance bound (5). The experimental performance of the universal algorithm, indicated by $\circ$s, is close to the bound.

In some regimes, the optimized baseline algorithm does quite well, in particular when ρ^{-1} is small (i.e., the queue is busy). If ρ is known in such situations, the computational simplicity of the baseline algorithm can be exploited with little loss in performance. If ρ^{-1} is smaller (the queue is more busy) than the baseline algorithm is designed for, distortion is dominated by buffer overflow. Conversely, if ρ^{-1} is bigger (the queue is less busy) than the designer thought, distortion is dominated by quantization noise. The performance of the baseline algorithms depends markedly on knowledge of ρ. If the value of ρ is uncertain or changing, the sample-path optimality of the universal algorithm is particularly useful.

The results of this section suggest a hybrid algorithm. Consider an algorithm that varies the fidelity at which new signals are stored depending on the observed history of queue overflows. If many 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 is controlled as a function of the empirical probability of buffer overflow to try to stay close to the performance of the optimized baseline algorithm. This strategy is akin to the approaches of [3], [4], and [8]. However, the gap between

the optimal baseline and the universal algorithm seen in Fig. 2 demonstrates that more dynamic distortion control is required to maximize system performance.

VI. Concluding Comment

In this note, we focus on signals that share a common distortion measure. As mentioned in Section II, the universal policy is directly applicable to situations where the distortion measure is signal specific. This is 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 them. Distortion measures can be designed 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 remain sample-path optimal.

References

[1] M. Alasti, K. Sayrafian-Pour, A. Ephremides, and N. Farvardin, "Multiple description coding in networks with congestion problem," *IEEE Trans. Inf. Theory*, vol. 47, no. 3, pp. 891–902, Mar. 2001.

[2] W. H. R. Equitz and T. M. Cover, "Successive refinement of information," *IEEE Trans. Inf. Theory*, vol. 37, no. 3, pp. 269–275, Mar. 1991.

[3] N. Farvardin and J. W. Modestino, "Adaptive buffer-instrumented entropy-coded quantizer performance for memoryless sources," *IEEE Trans. Inf. Theory*, vol. IT-32, no. 1, pp. 9–22, Jan. 1986.

[4] D. D. Harrison and J. W. Modestino, "Analysis and further results on adaptive entropy-coded quantization," *IEEE Trans. Inf. Theory*, vol. 36, no. 9, pp. 1069–1088, Sep. 1990.

[5] K. Ramchandran and M. Vetterli, "Multiresolution joint source-channel coding," in *Wireless Communications: Signal Processing Perspectives*, H. V. Poor and G. W. Wornell, Eds. Upper Saddle River, NJ: Prentice-Hall, 1998, ch. 7, pp. 282–329.

[6] B. E. Rimoldi, "Successive refinement of information: Characterization of the achievable rates," *IEEE Trans. Inf. Theory*, vol. 40, no. 1, pp. 253–259, Jan. 1994.

[7] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

[8] D. Tse, R. Gallager, and J. Tsitsiklis, "Optimal buffer control for variable-rate lossy compression," in *Proc. 31st Allerton Conf. Communication, Control, and Computing*, Monticello, IL, Sep. 1993, pp. 727–736.

On Centralized Optimal Control

Yu-Chi Ho

***Index Terms*—Decentralization, feedvack control, optimality.**

It can be argued that the Holy Grail of control theory is the determination of the optimal feedback control law or simply the feedback control law.[1] This is understandable given the huge success of the linear quadratic Gaussian (LQG) theory and applications for the past half-century. It is not an exaggeration to say that the entire aerospace industry, from the Apollo moon landing to the latest global positioning system (GPS), owe a debt to this control-theoretic development in the late 1950s and early 1960s. As a result, the curse of dimensionality notwithstanding, finding the optimal control law for more general dynamic systems remains an idealized goal for all problem solvers. We continue to hope that with each advance in computer hardware and mathematical theory, we will move one step closer to this ultimate goal. Efforts such as feedback linearization and multimode adaptive control [3] and [4] can be viewed as such successful attempts.

It is the thesis of this note to argue that this idealized goal of control theory is somewhat misplaced. We have been seduced by our early successes with the LQG theory and its extensions. The simple but often not emphasized fact is this: ***It is extremely difficult to specify and impossible to implement a general multivariable function even if the function is known***

Generally speaking, a one variable function is a two-column table, a two-variable function is then a book of tables, a three-variable function is a library of books, a four-variable function is a universe of libraries, and so on. Thus, how does one store or specify a general arbitrary 100-variable function, nevermind implementing it even if the function is God given? No amount of hardware advances will overcome this fundamental impossibility even if mathematical advances provide the general solution. Exponential growth is one law that cannot be overcome in general. Our earlier successes with LQG theory and its extensions were enabled by the fact that the functions involved have a very special form, namely, they decompose into sums or products of functions of single variables or low dimensions. As we move from the control of continuous-variable dynamic systems into discrete-event systems or more complex human-made systems such as electric power grids, communication networks, huge manufacturing plants, and supply chains, there is no prior reason to expect that the optimal control law for such systems will have the convenient additive or multiplicative form. Even if in the unlikely scenario that we are lucky enough to have such a simple functional form for the control law of the systems under study,

Manuscript received August 26, 2004. Recommended by Associate Editor E. Bai. This work was supported in part by the U.S. Army Research Office under Contract DAAD19-01-1-0610, by the U.S. Air Force Office of Scientific Research under Contract F49620-01-1-0288, and in part by the National Science Foundation under Contract ECS-0323685.

The author is with the Harvard University, Cambridge, MA 02138 USA, and also with the Center for Intelligent and Networked Systems (CFINS), Tsinghua University, Beijing 100084, China (e-mail: ho@hrl.harvard.edu).

Digital Object Identifier 10.1109/TAC.2005.844898

[1]Also known as decision rule, IF–THEN table, fuzzy logic, learning and adaptation algorithm, strategies, and a host of other names. However, nothing can be more general than the definition of a function that maps all available information into decision or action.