Deterministic Packet Marking for Congestion Price Estimation

R.W. Thommes and M.J. Coates
Department of Electrical and Computer Engineering
McGill University
3480 University St
Montreal, QC, Canada H3A 2A7
Email: rthomm@tsp.ece.mcgill.ca, coates@ece.mcgill.ca

***Abstract*— Several recent price-based congestion control schemes require relatively accurate path price estimates for successful operation. The proposed addition of the two-bit Explicit Congestion Notification (ECN) field in the IP header provides routers with a mechanism for conveying price information. Recently, two proposals have emerged for probabilistic packet marking at the routers; the proposals allow receivers to estimate path price from the fraction of marked packets. In this paper we introduce an alternative deterministic marking scheme for encoding path price. Under our approach, each router quantizes the price of its outgoing link to a fixed number of bits. We then make use of the IP identification (`IPid`) field to map data packets to different probe types, and each probe type calculates a partial sum of the path price bits. A router deduces its marking behaviour according to the `IPid` and the TTL (Time To Live) field of each packet. We evaluate the performance of our algorithm in terms of its error in representing the end-to-end price, and compare it to probabilistic marking. We show that based on empirical Internet traffic characteristics, our algorithm performs better when estimating path price using small blocks of packets. We also derive the probability distribution of the error for our scheme, and provide a relatively simple bound on its maximum mean-squared error.**

Keywords: statistics, network measurements, packet marking, congestion control

I. Introduction

In recent years, a variety of optimization-based network congestion control strategies have been proposed [1]–[7]. The majority of these are *price*-based congestion control protocols. They require that each router maintain a congestion price that is dependent on the flow into and out of its buffer. The routers must convey information about these prices back to the source so that its rate can be adjusted accordingly. The proposed addition of the two-bit Explicit Congestion Notification (ECN) field in the IP header provides routers with a mechanism for conveying price information [8]. In several of the proposals [2], [6], [7], the task of calculating link price is separable from the communication task (or packet marking exercise). Two probabilistic marking proposals have emerged for this case [2], [9]; the proposals allow receivers to estimate path price (the sum of link prices) from the fraction of marked packets. In this paper we introduce and evaluate an alternative deterministic marking scheme for encoding and estimating path price.

A. The Marking Problem

The problem we examine in this paper is that of determining the sum of the prices of a set of links constituting a path between a source and a receiver. We now formalize the problem as in [9], slightly deviating occasionally. Consider a set of links $1, \ldots, n$ forming an end-to-end path from a source to a receiver. Associated with each link i is a non-negative price s_i. Let $z_n = \sum_{i=1}^{n} s_i$ denote the sum of the prices along the path. The routers convey congestion information to the receiver by encoding this information in the data packets that traverse the path. For each packet, the current IP standard provides a router with access to two congestion notification (CN) bits in the IP header. We denote the state of these bits after the marking action of link/router i by X_i. X_i can take on four possible values $00, 01, 10, 11$, but one state (00) must be reserved to communicate the inability to use ECN, so three states are available for information passing (although implementation of RFC 3540 [10] would effectively reduce this to two). Upon receipt of the marked data packets, the receiver must be able to form an estimate of z_n, and feed this estimate back to the sender. In this paper, we focus primarily upon the problem of routers conveying congestion information to the receiver. Our goal is to design a *marking algorithm* that the routers can apply to all the packets that traverse them, with the algorithm potentially modifying the IP congestion notification header bits and thus providing congestion information to the receiver.

As discussed in [9], a marking algorithm must obey some design constraints. The algorithm must be fully distributed, so that in making a marking decision, each router makes use of only local information – specifically, the current link price and the information contained in the IP header of the packet. The algorithm should not retain per-flow state nor retain any memory of how previous packets were marked. Adler et al. [9] claim that "there is no deterministic marking algorithm under these conditions". They derive a probabilistic algorithm, Random Additive Marking (RAM) and compare its behaviour to that of Random Exponential Marking (REM) which was proposed by Athuraliya et al. [2]. Significant variance in price estimates persists in the probabilistic schemes despite the use of a substantial number of data packets. Such a property is unavoidable, because a binomial probability lurks

beneath any probabilistic scheme, and one can at best achieve a linear decay in variance or mean-squared error. Packet prices are dynamic, so accurate estimates must be obtained with a relatively small number of packets. Moreover, the estimation error is highly dependent on the actual value of the link prices; the garnering of accurate estimates in RAM requires that the mean link price be close to 0.5 (assuming that prices are bound between zero and one) [9]. The performance of REM is heavily contingent on the setting of its parameters and how well they match the proffered prices and path lengths [9].

We beg to differ with Adler's claim, and our paper focuses on outlining a deterministic quantized marking (DQM) algorithm and exploring its performance. In developing the deterministic scheme, we have attempted to introduce as few changes to the current protocols as possible, in order to make deployment a more attainable objective. The scheme does not demand changes to the TCP or IP headers, but it does require that the available congestion notification bits are used in a different manner. The major change is that a completely different, but simple and computationally inexpensive, marking algorithm must be implemented at the routers constituting a path. The majority of the paper focuses on developing and analysing a method for conveying congestion information on the forward path from sender to receiver.

The paper is organized as follows. In Section II we review the current IP standard for packet marking and examine in more detail the probabilistic marking algorithms, REM and RAM. In Section III we propose a new deterministic marking algorithm and describe its structure. In Section IV we analyse the performance of the deterministic algorithm in terms of mean-squared error and make comparison with RAM and REM. In Section V we examine the performance of our algorithm based on Internet trace-driven simulations.

II. Packet Marking and Probabilistic Techniques

A. IP Standard

RFC 3168 proposes the addition of Explicit Congestion Notification (ECN) to the IP protocol [8]. ECN provides an alternative to the current method by which intermediate network routers signal congestion to end systems by dropping packets. Under the new proposal, a two bit ECN field in the IP header may be used to indicate the presence of congestion. In the proposal, the two bits are considered to comprise four *codepoints*. For backwards-compatibility, RFC 3168 specifies that one codepoint (00) be used to indicate that the packet is not using ECN.

When ECN is used, the sender initially sets the ECN bits to either one of the two codepoints (01,10) to indicate ECT (ECN-Capable Transport), which implies that it and the intended receiver support the use of ECN. When an intermediate node experiences congestion and receives a packet with an ECT codepoint, it sets the ECN field to the Congestion Experienced (CE) codepoint (11) to signal the congestion to the receiver. The RFC requires that when an end system receives a single packet with the CE codepoint set, its transport layer protocol must respond in essentially the same manner as when a dropped packet is detected.

The RFC does not specify the metric used by a router to calculate its level of congestion nor how it should determine whether to mark a given packet.

B. REM

Athuraliya and Low proposed the Random Exponential Marking (REM) scheme in [2]. The price of each link is constrained to be non-negative, but is unbounded. Initially the ECN codepoint X_0 is set to 01 (10 could also be used). At the i-th router, if $X_{i-1} = 01$, REM sets the ECN codepoint to 11 with probability $1 - \phi^{-s_i}$. Here $\phi > 1$ is a parameter chosen by the designer. If an incoming packet already has its ECN codepoint set to 11, the router passes it on unchanged. At the receiver, the value of the codepoint is X_n. $X_n = 01$ with probability $\phi^{-\sum_{i=1}^{n} s_i}$ and $X_n = 11$ otherwise. The expectation of the indicator function $\mathcal{I}[X_n = 11]$ is $1 - \phi^{z_n}$. An estimate can be formed of the total price by collecting N packets, computing $\overline{X} = \sum_{j=1}^{N} \mathcal{I}_j[X_n = 11]$ and forming the estimate $\widehat{z_n} = -\log_\phi(1 - \overline{X})$. As pointed out in [9], this estimate is biased, but does converge to z_n almost everywhere, almost surely. The local computation requires no information aside from the link price, but the choice of ϕ is difficult. Adler et al. analyse the performance of REM [9], demonstrating that it is highly dependent on the value of ϕ. The optimal choice of ϕ depends on the path length and the end-to-end price, which means that it generally cannot be deduced. Suboptimal choices of ϕ can lead to very poor estimation performance, arising primarily when link marking probabilities hover near the extreme values of 0 or 1 [9].

C. RAM

Adler et al. propose an alternative scheme, Random Additive Marking (RAM) [9]. RAM imposes the additional requirement that the price of each link can be bounded between 0 and 1, and that each router knows its position i within the path. The authors describe a method for estimating this position based on the time-to-live (TTL) field of the IP header and demonstrate that price estimation performance does not substantially degrade through its application. The RAM algorithm is described in terms of a single bit, but here, for consistency with our notation, we provide an equivalent four state description. Initially $X_0 = 01$. The i-th router in the path leaves the ECN field X_{i-1} unchanged with probability $(i-1)/i$, sets it to 11 with probability s_i/i, and sets it to 01 otherwise. The expectation of the binary random variable $\mathcal{I}[X_n = 11]$ is equal to $\sum_{i=1}^{n} s_i/n$. Collecting N data packets and computing the average of the indicator function (multiplied by the path length n) produces an unbiased estimate of the path price.

Adler et al. conduct a performance comparison between RAM and REM under the assumption of independent, uniformly distributed link prices that are normalized so that the mean link price is approximately 0.5. In terms of error probability, RAM always outperforms REM even when the optimal value of ϕ is used. Furthermore, unlike REM, the

performance of RAM is independent of the path length. However, the performance of RAM suffers severely when average link price strays substantially away from 0.5.

D. Potential REM and RAM Extensions

Our deterministic algorithm makes use of three ECN codepoints to encode path price information, whereas REM and RAM only use two unique codepoints. It should be possible to modify REM and/or RAM to take advantage of a third codepoint in order to provide better price estimation. However, this would require substantial changes to the algorithms and we do not explore the issue further in this paper.

III. Deterministic Packet Marking

A. Preliminaries

We now outline a marking mechanism that allows the routers lying on a path between host and client to convey the sum of their quantized prices to the host. The algorithm we propose makes use of only the two existing ECN bits in the IP header, but modifies both the manner in which routers perform marking and also the manner in which the client interprets the marking information. In contrast to the probabilistic marking schemes discussed above, the marking scheme we now outline is deterministic in the sense that *every* packet is marked and the nature of the mark is derived via a deterministic function applied to quantized link prices. Our proposed marking procedure retains the RFC 3168 allocation of the 00-codepoint as an indication that a packet belongs to a stream that does not support ECN [8].

Our scheme requires (in the same fashion as RAM) that every link price s_i is bounded $0 \leq s_i \leq 1$. Every router applies a uniform quantizer to its price, making use of b bits. In the examples we provide in this paper and in the empirical analysis, we make use of $b = 4$, because this value provides a good balance between quantization error and the number of bits needed to describe the price estimates.

The key idea behind the scheme is that each data packet calculates the sum of a small subset of the link price bits. By combining all of the partial sums, the receiver can reconstruct the sum of quantized prices and form an estimate of the path price. Figure 1 illustrates the idea; data packets are mapped into *probe types*, and each probe type adds two bits together; the bits occupy the same degree of significance and occur in different routers. As an example from Figure 1(a), probe type 7 adds the most significant bits (MSBs) of the quantized prices of links 3 and 4. The output of the sum is indicated in Figure 1(b). The summation procedure is initialized by setting X_0 to 01. After that, only designated links can make a change to the state; in this example, the designated links are 3 and 4. These links change the state if the link price bit to which the probe type corresponds (the most significant bit in our example) is equal to 1. If that is the case, the state is updated according to the sum table; i.e., state 01 is changed to state 10, and state 10 is changed to 11. When the data packet arrives at the receiver, it indicates the sum of the two bits, as depicted in Figure 1(b).

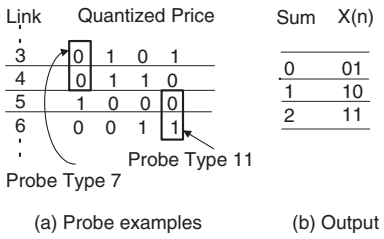


Fig. 1. Example of the operation of Deterministic Marking Algorithm I. (a) The nature of the probe types – each performs a sum of two bits, as indicated by the boxes. (b) The state output of the sum.

In order for such a procedure to work, we need a means of labelling each data packet as a certain probe type. The routers must also be able to determine how to react when encountering a particular probe type. We perform the packet labelling exercise by utilising the IP identifier field. The decision to mark at the routers is based on the time-to-live (TTL) field of the packet and its probe type label. The number of probes we need is equal to $b\lceil n/2 \rceil$, where n is the number of links and b is the number of quantization bits. It is important that we restrict the number of probe types to a reasonable number. For this reason, our algorithm imposes the restriction that n is bounded, $n \leq n_{\max}$. For the purpose of a concrete description of our technique, we use $n_{\max} = 30$ in this paper, which implies a need for 60 probe types. Later in this section, we will demonstrate how this requirement can be reduced to 40 probe types. The results of Begtavesic et al. [11] indicate that paths of length greater than 30 are very rare in the Internet. In the rare event that a path does contain more than 30 links, the results are not catastrophic, but there is a small probability of overflow error.

B. The IP identification field and Probe Types

The purpose of the IP identification (`IPid`) field is to provide a mechanism for fragment reassembly. RFC 791 [12] states that it "is used to distinguish the fragments of one datagram from another" and that for each datagram, the identification field must be set to "a value that must be unique for the source-destination pair and protocol for the time the datagram will be active in the Internet system." Beyond this, there are no requirements placed on the actual value of the `IPid` field.

Bellovin [13] describes a technique for counting NATted hosts that exploits the manner in which many hosts implement the `IPid` field. He makes the observation, also noted in [14], that many hosts implement the `IPid` field using a simple counter. That is, successive data packets emitted by a host carry sequential `IPid` fields. There are exceptions; some hosts use byte-swapped counters, and others use pseudo-random number generators [13]. Some versions of Solaris use separate sequence number spaces for each (source, destination, protocol) triple [13].

The uniqueness of the `IPid` field makes it a natural candidate for use as a probe type identifier. The field is 16 bits, and we only require $b\lceil n/2 \rceil$ (set to 60 in our example)

probe types, so we need to develop a mapping function. In constructing the mapping function, one of our major goals is that each probe type appears as regularly as possible: ideally once every 60 packets. (In fact, estimation performance can be improved by enforcing a concentration on the more significant bits.) Here the counter implementation, and particularly the Solaris mechanism of separate sequence number spaces, is of substantial benefit. We choose a prime number m slightly larger than our required number of probe types, and set the probe-type identifier to `IPid` $\bmod m$. This produces m probe types; the majority are used for price inquiries, and the remainder are reserved for other forms of communication.

If the host implements the `IPid` field as a counter, a sequence of data packets between a given host and receiver cycles through the probe types, skipping some values whenever packets belonging to alternative streams intervene. With this implementation, each probe type appears very regularly. The use of a prime number m addresses the scenarios of pseudo-random `IPid` generation and byte-swapped counters. In the following section, we analyse the performance of our marking algorithm assuming random `IPid` generation. Figure 3(a) in Section V displays an empirical survival function of the spacing between probe types of the same kind using data collected from the Internet. The figure illustrates that in most cases, the spacing is slightly larger than m, indicating the prevalence of counting implementations in the Internet. A strict random generation results in a substantially heavier tail.

In the ideal scenario, a separate space is dedicated to each (source, destination, protocol) triple. The `IPid` field is then generated by a counter, and every data packet belonging to that triple is used to calculate the path price. This ensures that (in the absence of lost packets) each probe type appears once every m data packets.

C. The router marking algorithm

We now outline the marking mechanism used by each router. We outline two algorithms. The first, Algorithm I, uses a simple bit summation strategy and allows us to focus on the marking technique. The second, Algorithm II, uses a more complicated technique to generate price bit summations, but results in a substantial saving in the number of required probe types.

When a data packet arrives at a router, the router calculates its Link ID, which is a packet-specific value. This is determined as $\mathrm{LinkID} = \texttt{TTL} \bmod n_{\max}$. The router also determines the probe type of the packet: $\mathrm{ProbeType} = \texttt{IPid} \bmod m$. The pair (Link ID, Probe Type) determines whether router i should perform any action and, if so, which price bit it should focus on. If the indicated price bit is 1, router i increments the state X_{i-1}, from 01 to 10 or from 10 to 11. In Algorithm I, each probe type requires an action for only two Link IDs and is associated with a specific bit significance. Figure 1 depicts example probes and the nature of the output.

Algorithm II performs the bit summation in a different manner. Figure 2 depicts its operation. Each probe focuses on six bits of the same significance (examples are shown in Figure 2(d)). However, in this case the mechanism for determining a state transition is slightly more complicated. Each probe type is associated with one of two categories, A or B. If the (Link ID, Probe Type) pair indicates that router i should participate, it checks the indicated price bit. If the price bit is zero, it takes no action, setting $X_i = X_{i-1}$. If the price bit is one, the router sets a new value of X_i. This value is determined by the category of the probe and the associated state transition diagrams, eiher that of Figure 2(a) or that of Figure 2(b). Two probe types of different categories focus on the same six bits. The output state of the two probe types uniquely determines the sum of the six bits, as depicted in Figure 2(c). As two probe types are sufficient to identify the sum of six bits, the number of probe types required to cover all bits is reduced by a factor of 2/3 to $2b\lceil n/6 \rceil$. In our example of $b = 4$ and $n_{\max} = 30$, we require 40 probe types and use $m = 43$ as the generator of the probe type.

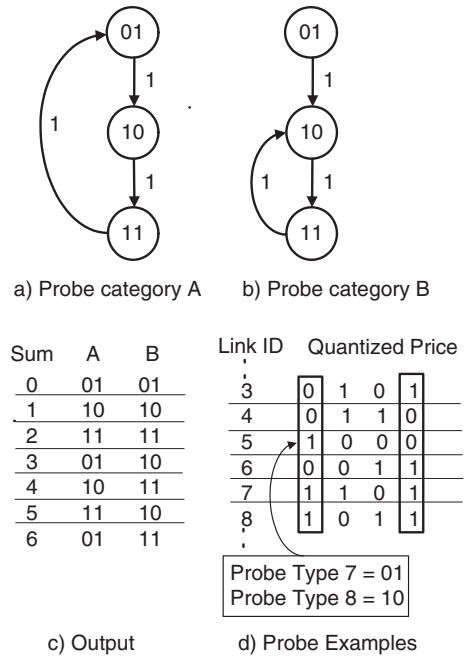


Fig. 2. Algorithm II operation. (a) and (b): State transition diagrams for probe categories A and B, respectively. (c) The output of probe categories A and B depending on the sum of the bits. (d) Examples of probe types.

D. Path Price Estimation

When all of the probes have been collected, the sum of quantized prices can be determined exactly, and the estimation error is equal to the quantization error (as detailed in the following section). However, in any practical scenario, the price can only be assumed to remain fixed for a short period of time. We specify two procedures for estimation in such a scenario:

1) *Block-based estimation*: For each block of K data packets, form an estimate based on the set of available probe types. If probe types are missing, insert the values that

minimize the expected error under the assumption of a uniform distribution on link price values.

2) *Time-varying estimation*: A new estimate is formed upon the reception of every data packet based on the current set of values for all probe types.

In the following section, we provide bounds on the mean-squared error under the block-based estimation approach. In Section V we analyse performance empirically using probe type sequences derived from Internet traces.

IV. Error Analysis

In this section, we analyse the performance of our marking algorithm based on fixed block lengths, with probe types generated independently and uniformly over the possible m values. Under the assumption that the quantization error is uniformly distributed between its positive and negative extrema (note that this is the case when link prices are uniformly distributed over $[0, 1]$), we develop an expression for the probability density of the path-price estimation error in the deterministic marking algorithm and bound the mean-squared error. In Section V we perform an empirical analysis of mean-squared error for comparison.

We wish to emphasize that our above assumption of the distribution of quantization error is important to determining the performance of our algorithm, since one can identify scenarios where our algorithm will perform poorly. A pathological example is the case where each link price is very slightly smaller than the midpoint of two arbitrary representation points, such that the quantization error is consistently negative. Such a scenario is highly unlikely, but a more important scenario in practice is the situation where the majority of link prices are extremely small. The latter case is likely to arise if TCP is used as the congestion control algorithm and there is a straightforward mapping of path price to marking probability [2]. We do not analyse such a scenario in this paper, but suspect that REM, suitably parameterized, is better tailored to the situation because of the exponential aspect of its marking behaviour.

There are two potential sources of error when a receiver calculates the end-to-end price using the deterministic marking algorithm. One is due to the quantization, and the other is due to the possibility of one or more unique probes types not being observed. This section of the paper derives the distribution of the error. We assume that the path is of length n links, that each link's price is quantized using b bits, that a block size of K packets is examined by the receiver to determine the end-to-end price, and that the encoding scheme uses 1 packet to encode 2 bits, meaning that the number of unique probe types m is $\lceil \frac{nb}{2} \rceil$. We commence with some background discussion.

1) Quantization Error: Our deterministic algorithm scheme adopts a uniform b-bit quantizer for link prices normalized to fall in the interval $[0, 1)$. That is to say, its representation points will of the form

$$\left\{ \frac{1}{2^{b+1}}, \frac{3}{2^{b+1}}, \frac{5}{2^{b+1}}, ..., \frac{2^{b+1}-1}{2^{b+1}} \right\} \quad (1)$$

The quantization error e_q is uniformly distributed between the two extrema: $f(e_q) \sim U\left(\frac{-1}{2^{b+1}}, \frac{1}{2^{b+1}}\right)$ and has a variance of $Var(e_q) = \frac{1}{12 \cdot 2^{2b}}$.

2) Error due to Missing Bits: In order to examine the error in representing the price of a link due to missing one or more of the b bits, we consider the decimal representation q_d of a binary quantization value $(a_0, a_1, ..., a_{b-1})$ where a_0 is the MSB (most significant bit). It is given by:

$$q_d = \frac{1}{2^{b+1}} + \left(1 - \frac{1}{2^b}\right)\left(\frac{a_0 2^{b-1} + ... + a_{b-1}2^0}{2^b}\right) \quad (2)$$

For the purposes of estimating q_d we adapt the approach of replacing a missing bit a_i by its expected value under the assumption of a uniform distribution. Since any bit is equally likely to be 0 or 1, a missing bit will always be replaced by a value of $1/2$. Clearly this results in a lower error variance than if a missing bit is randomly chosen to be 0 or 1. The error due to a missing bit a_i is a discrete uniform random variable (denoted by U_D) taking on one of two values with equal probability:

$$e_{a_i} \sim U_D \left\{ -\left(1 - \frac{1}{2^b}\right)\frac{1}{2^{2+i}}, \left(1 - \frac{1}{2^b}\right)\frac{1}{2^{2+i}} \right\} \quad (3)$$

The expected value of the error e_{a_i} is 0, and its variance is given by: $Var(e_{a_i}) = \frac{(1-\frac{1}{2^b})^2}{12 \cdot 2^{2i+2}}$

A. Distribution of Missing Probe Types

The distribution of the number of probes not observed in a block of size K is strongly dependent on how the IP identifier field increments between contiguously transmitted packets. Ideally, the identifier increases by one every time, in which case any block of $K > m$ is guaranteed to include at least one packet mapping to every probe type. However, it is known (and verified by empirical data) that some hosts on the Internet generate IP identifiers that change in a random manner. In this section, we consider the case where identifiers and hence probe types are randomly selected based on a uniform distribution. We note that strictly speaking it is not difficult to identify cases worse than randomly distributed identifiers. For instance, if all identifiers corresponding to a certain value $s \bmod m$ are skipped by the host, then the probe type corresponding to this value will never be observed for any sequence length. However, our observation of Internet traffic does not provide evidence that such pathological behaviour is likely to occur.

Due to our assumption of all probe types being equally likely, the analysis of the number of missing probes is an instance of the classical occupancy problem [15], [16]. Briefly, this problem considers the number of empty bins resulting from a random allocation of K balls into m different bins.

B. Representing the End-to-End Error

Consider an estimation block of K packets in a scenario where there are m unique probe types. Define the random variable $W : \min(1, K) \leq W \leq m$ as the number of probe types not observed in the block, and let $\mathbf{w}$ be a vector of the form $(w_0, w_1, ..., w_{b-1})$ where $w_i : 0 \leq w_i \leq n/2$ is how

many probe types encoding information about the ith price bit-column have not been observed.

Theorem 1: If quantization is performed on each link price using b bits and there are n links, a block of K probes uniformly generated from m possible probe types is used to perform path-price estimation, and the quantization error is uniformly distributed between extrema, then the probability density of the estimation error e_t is:

$$p(e_t|n,b,m,K) = \sum_{W=\max(0,m-K)}^{m-1} p(W|m,K) \times \sum_{\mathbf{w}:\sum w_i=W} p(\mathbf{w}|W,n,b)\cdot p(e_t|\mathbf{w},n,b) \quad (4)$$

Here $p(W|m,K)$ is defined below by (5), $p(\mathbf{w}|W,n,b)$ by (6), and $p(e_t|\mathbf{w},n,b)$ by (7).

The decomposition of (4) is readily verifiable using the Law of Total Probability. We complete our derivation of the density function for the end-to-end error by considering each of the three component conditional distributions defined in Theorem 1 in turn.

The result of interest arising from the Occupancy Problem is the distribution of the number of missing entities when drawing a certain sample size [15]. This result may be directly applied to establish the distribution for W:

$$p(W|m,K) = \binom{m}{W}\sum_{i=0}^{m-W}(-1)^i\binom{m-W}{i}\left(1-\frac{W+i}{m}\right)^K \quad (5)$$

Our next objective is to determine the distribution of the components of the vector $\mathbf{w}$ given that W total probes are not observed. Each of the b price bit-columns has an equal proportion of the total number of probes m assigned to encode it, so a missing probe is equally likely to be from any one of the b columns. Thus, the distribution of the vector $\mathbf{w}$ is multivariate hypergeometric with $w_i \le \frac{n}{2}$ and $\sum_{i=0}^{b-1} w_i = W$:

$$p(\mathbf{w}|W,n,b) = \frac{\prod_{i=0}^{b-1}\binom{n/2}{w_i}}{\binom{m}{W}} \quad (6)$$

Finally, we consider the distribution of the error in the end-to-end price given $\mathbf{w}$. The error is the combined sum of quantization errors of the price of each link and the errors due to missing bits. The distribution is comprised of a discrete component due to missing bits, and a continuous component arising from the quantization errors. The distribution of the total quantization error is the n-fold convolution over n IID uniform random variables. It has a mean of 0, is distributed in the interval $\left[-\frac{n}{2^{b+1}}, \frac{n}{2^{b+1}}\right]$, and has a variance of $\frac{n}{12\cdot 2^{2b}}$. We will denote a shifted version of this distribution centered at μ as g_μ.

The distribution due to the $2W$ missing bits may be obtained by evaluating the error arising from every one of the 2^{2W} cases. Each case has probability $\frac{1}{2^{2W}}$, meaning that it is simply a matter of identifying all the possible error values and determining how many different cases map to each value. Let the total number of possible error values be v, and construct a $(2\times v)$ probability matrix P in which entry $P_{1,i}$ identifies the i-th error value and $P_{2,i}$ is its probability. The distribution of the total error due to quantization and missing bits is then given by the following sum:

$$p(e_t|\mathbf{w},n,b) = \sum_{i=1}^{v} P_{2,i}\cdot g_{P_{1,i}}(e_t) \quad (7)$$

C. A Mean-Squared Error Bound

We now derive a bound on the mean-squared error of the end-to-end price estimate assuming uniform generation of probe types. We make use of another result related to the Occupancy Problem – an upper bound $H(K,m,W)$ on the probability $p(W|m,K)$ of missing W probe types when K probes are generated and there are m unique types (from Theorem III in [17]). Specifically:

$$H(K,m,W) = \exp\left[-\left(W\ln\left(\frac{W}{E[W]}\right)-W-E[W]\right)\right] \quad (8)$$

where $E[W] = m\left(1-\frac{1}{m}\right)^K$.

Now, given that W probe types are not observed, the scenario that maximizes the contribution to the total error is if all missing probes are of the types measuring the MSB of the quantized price. Thus, we will assume that $2W$ MSBs are unobserved. If $2W$ exceeds n, some of the missing probes must correspond to non-MSB bits. However, we will model all missing probes as contributing an error equal to the maximum. This is in keeping with deriving an upper bound on total error variance.

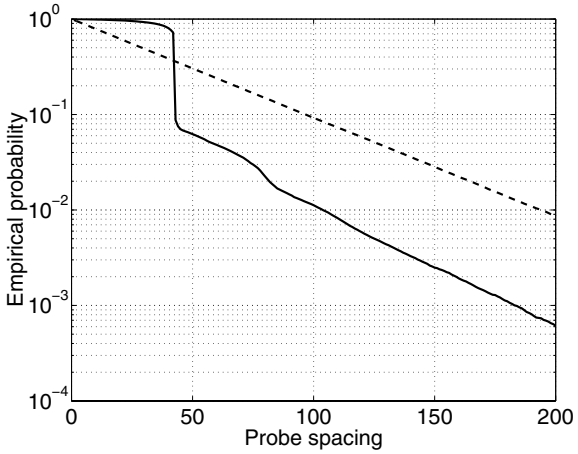
Each unobserved probe results in two MSBs being replaced by their expected value of $1/2$ for purposes of calculating the quantized price for a given link. Given that an MSB a_0 contributes $\left(1-\frac{1}{2^b}\right)\left(\frac{a_0 2^{b-1}}{2^b}\right)$ to the quantized price, a missing MSB results in a discrete error taking on the values $\pm\left(1-\frac{1}{2^b}\right)\left(\frac{1}{2^2}\right)$ with equal probability. We can thus upper-bound the error due to a missing MSB (and hence any bit) at $\pm 1/4$ and the variance at $1/16$.

Finally, due to independence, the variance in the error of the path price estimate is just the sum of the individual variances, given by $\frac{n}{12\cdot 2^{2b}} + \frac{W}{8}$.

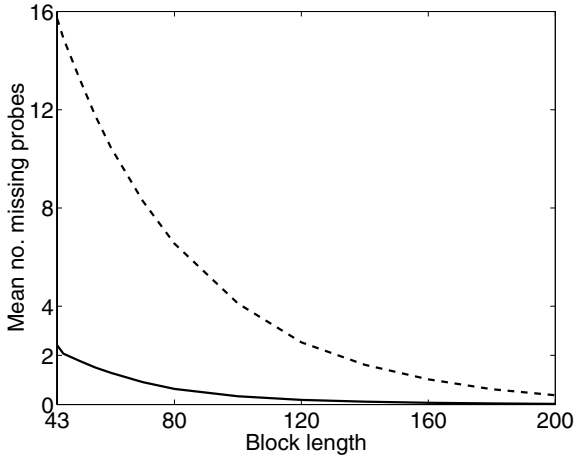
Multiplying this variance by (8) and summing over all possible values of W then gives the following result:

Theorem 2: For an estimation block of K probes uniformly drawn from m unique probes, with n links in the path, b bits used to quantize each link price, and a uniform distribution of the price quantization error, the mean-squared error in the path-price estimation is bounded as:

$$MSE(m,b,K,n) \le \sum_{W=0}^{K-1} H(K,m,W)\frac{W}{8} + \frac{n}{12\cdot 2^{2b}} \quad (9)$$



(a) The survival function of the spacing between probes of the same type.



(b) The mean number of missing probes as a function of block length.

Fig. 3. Analysis of the distribution of probe types (when 43 probe types are defined). In each plot, the solid curve corresponds to the Internet traces and the dashed curve to uniform random allocation of probe type.

where $H(K, m, W)$ is determined from (8).

V. Simulation Performance

In this section of the paper, we analyse the performance of the estimation algorithm using trace-driven simulations. We collected 100 traces of 2000 packets by downloading files from 100 different servers. Of these servers, 50 were based in USA/Canada, 25 in Europe, 15 in Asia and the remaining 10 scattered across the globe. We extracted the IP identifier field for each packet, and mapped the field into a probe type using $m = 43$ probes. We perform our analyses assuming a maximum path length of 30 (the TTL is taken modulo 30) and 4 bit quantization. These settings imply that 40 probe types are needed for estimation, if Algorithm II is used. The remaining three are reserved for protocol communication.

A. Missing Probe Types

First, we examined the nature of the distribution of probe types in the empirical data sequences. It is important that every probe type appears regularly. Figure 3(a) shows the survival function of the spacing between probes of the same type over the 100 traces. The function generated from a uniform allocation of probe types is shown for comparison. If the IP identifier were implemented as a strict counter, with no intervening packets from the host, probe types of the same kind would always be 43 packets apart. The empirical survival function indicates that the counter implementation is widespread; more than 90 percent of the time, the spacing is 43 packets. Sometimes the probe types are spaced much further apart; this occurs due to pseudorandom implementations and busy servers where the counter frequently skips. Ninety-nine percent of the time, the spacing between probes of the same type is less than 100 packets. Clearly, the uniform generation of probe types results in a substantially heavier tail in the spacing distribution.

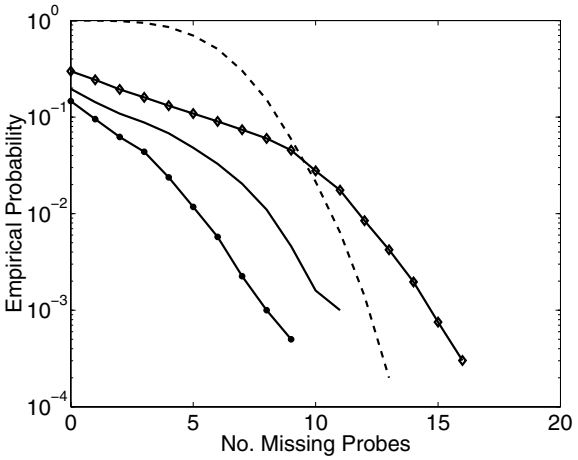


Fig. 4. The survival function for the number of missing probe types. Dashed line corresponds to uniform random probe type allocation for a block length of 80. Solid lines correspond to Internet traces; diamond marker is block length 60, no marker block length 80, * marker block length 100.

The second feature that interests us for the purposes of block-based path price estimation is the number of probe types missing from a block of size K. Figure 3(b) shows the mean number of missing probes as a function of block length for the empirical traces and for uniform random allocation. For a block length of 43, the mean number of missing probe types is slightly larger than 2 in the empirical case (as compared to 16 for uniform allocation). The effects of the IP identifier counter implementation are again evident. The mean number of missing probe types is substantially less than 1 when the block size is 80.

Figure 4 show the empirical survival functions for the number of missing probes in a block for the Internet traces. In the case of a block size of 80 packets, 80 percent of the time there are no missing types, and approximately 95 percent of

the time there are less than 5 missing types.

B. Error Analysis

We performed an examination of the error probability of the deterministic estimation to enable comparison with RAM [9] probabilistic price estimation. The error probability $err(\epsilon)$ is defined as the probability that the path-price estimate falls outside of a range (defined by ϵ) about the true price:

$$err(\epsilon) = 1 - \Pr[(1-\epsilon)z \leq \widehat{z} \leq (1+\epsilon)z] \qquad (10)$$

We examined two cases. In each case, the path consisted of 10 links. We modeled the link prices as mutually independent and dynamically varying. In case 1, the link prices were uniformly distributed over $[0, 0.2]$. In case 2, the link prices were uniformly distributed over $[0.4, 0.6]$. For each case, one hundred realizations of link prices were generated, and the error probability was evaluated for each realization using the Internet traces to generate probe types. The results were averaged to determine an average error probability.

Figure 5 displays the resulting averages for $\epsilon = 0.1$ as a function of block length. The performance of RAM is shown for comparison (with results derived from expressions in [9]). Both algorithms show improved performance when the marking range is centred around 0.5. Note, however, that the performance differential for the deterministic algorithm is due solely to the presence of the normalizing term in the error probability bounds. The intrinsic estimation accuracy, in raw terms, is identical in the two cases. The deterministic algorithm outperforms RAM in both cases for small block lengths when the Internet traces are used to generate probe types. However, RAM outperforms the deterministic algorithm for block sizes below 120 when probe types are generated from a uniform distribution. Figure 6(a) shows the error probability as a function of ϵ for a block-length of 100.

C. Mean-Squared Error Analysis

We performed an analysis of the mean-squared error of the deterministic algorithm for the case of uniformly distributed link prices over $[0, 1]$. We generated 100 realizations of link prices for a path of length 20 links, and then evaluated the mean-squared error for each realization. The probe types were generated both from the Internet traces and from a uniform random allocation. Figure 6(b) shows the normalized mean-squared error (the error is normalized by the number of links in the path). The mean-squared error performance of RAM is shown for comparison. As in error probability, the deterministic algorithm outperforms RAM for small block lengths when probe types are generated from the Internet traces. As the block length becomes large (> 1000), RAM begins to outperform the deterministic algorithm; the performance of the latter is bounded by quantization error.

VI. Practical Considerations and Future Work

A. Receiver Feedback

Any price-based congestion control protocol requires that the source is able to determine the price along its path to the receiver. As is the case for the REM and RAM proposals, we have not addressed the mechanism by which a receiver informs the sender of its path price estimate (the performance of REM in conjunction with RFC 3168-style feedback has been explored via simulation in [2]). RFC 3168 suggests the addition of a 1-bit ECN-echo (ECE) in the TCP header [8]. The RFC proposes that after a client receives a packet with the ECN field in the CE state, it set the ECE bit to 1 in every acknowledgement it sends until it receives a packet from the sender with the Congestion Window Reduced (CWR) flag set. The CWR flag, another new TCP flag suggested in [8], is set by the sender to acknowledge the receipt of an ECE packet.

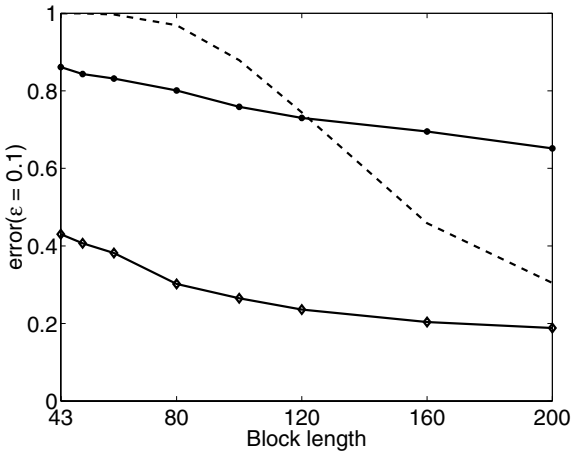
Unfortunately, this mechanism cannot be readily adapted to provide explicit feedback of the path price to the sender. The fundamental problem is that there is the potential for multiple acknowledgements to be sent with the ECE field set in response to a single received packet with the ECN field in the CE state. This means that the the sender is unable to accurately determine what proportion of its sent packets were marked. Thus, a novel approach is necessary in order to provide more informative price feedback.

B. Security

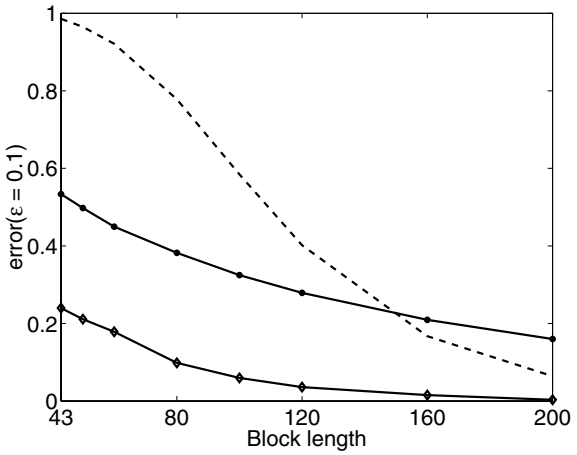
Closely tied to the issue of price feedback is the question of security. There are a number of scenarios which would result in a sender receiving erroneous price information if security concerns are not addressed. First, a malicious receiver stands to receive a disproportionate share of a congested link's capacity if it conceals the presence of congestion from the sender [18]. Second, a misconfigured or malfunctioning router may corrupt pricing information.

RFC 3540 outlines an addition to the proposed ECN standard which prevents the malicious or accidental erasure of congestion information [10]. The proposal exploits the fact that there are two ECN codepoints that indicate that a packet supports ECN marking but has not yet been marked. RFC 3540 proposes that, for each packet, a sender encode a randomly-selected 1-bit nonce using these two codepoints. The RFC defines a new 1-bit flag in the TCP header – the Nonce Sum (NS) – that is used by the receiver to carry the 1-bit sum of the nonces over the range of TCP data bytes being acknowledged by a given acknowledgement. If a router or the receiver attempts to clear a marked packet, it will have no knowledge of the original nonce. Thus, it will only have a 1 in 2 chance of correctly guessing the original nonce. An incorrect choice will result in an incorrect NS value being returned to the sender. The sender verifies all NS values, and thus will quickly detect persistent ECN erasures.

The addition of nonces is fully compatible with REM, and indeed alleviates security concerns. However, RAM allows for a router to legitimately "unmark" a packet, and thus will not function properly with the approach suggested above. Our deterministic algorithm requires three unique codepoints, and thus is also not compatible with RFC 3540. However, a modification to our Algorithm I will provide limited security against a malicious receiver concealing congestion. Rather

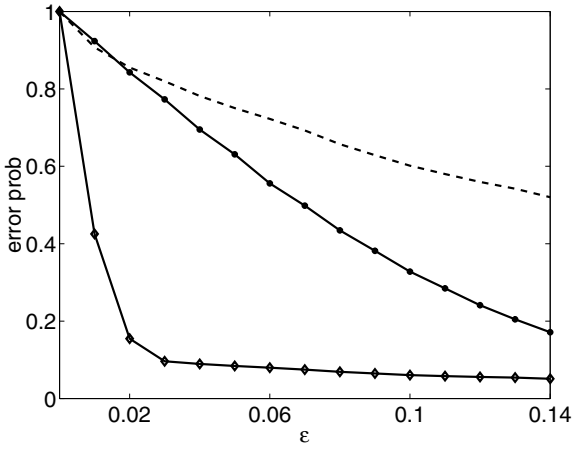


(a) Link prices uniformly distributed over the range 0 to 0.2

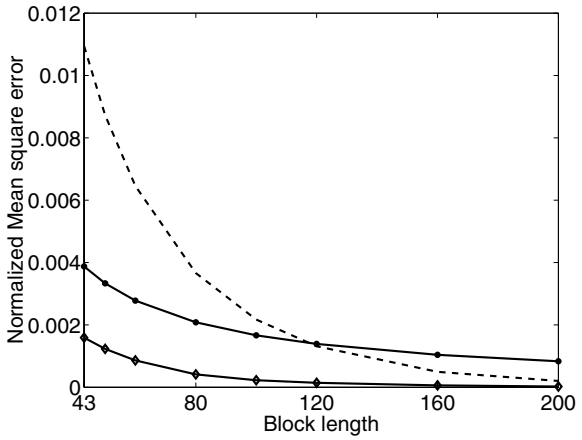


(b) Link prices uniformly distributed over the range 0.4 to 0.6

Fig. 5. Error probability as a function of block length for a path of 10 links and $\epsilon = 0.1$. The solid line with diamond markers corresponds to the probe types generated from the Internet trace and the deterministic algorithm. The * markers correspond to the RAM algorithm. The dashed line is the deterministic algorithm with uniform random probe type generation.



(a) The error probability as a function of ϵ for a block length of 100.



(b) Normalized mean-squared error as a function of block length. Path length of 20 links and prices uniformly distributed between 0 and 1. Error is normalized by the number of links (20).

Fig. 6. Error Analysis – error probability and mean-squared error. The solid line with diamond markers is the deterministic algorithm with probe types generated from Internet traces. The * markers are the RAM algorithm. The dashed line is the deterministic algorithm with uniform random probe type generation.

than always initializing the ECN field to 01, the sender randomly initializes it to any one of the 3 codepoints. Routers increment the field as before, but may now also "wrap-around" from 11 to 01. When the packet arrives at the receiver, it is not aware of the initial ECN field value and thus is unable to reliably (from its point of view) alter pricing information prior to providing feedback to the sender. This modification would, however, render receivers incapable of calculating path prices. They would have to transmit raw ECN field data from all received packets back to the sender, allowing it to use this information, along with knowledge of the packets' initial codepoints, to calculate the path price.

VII. Conclusion

We have specified a novel deterministic packet marking algorithm that allows a receiver to deduce the sum of the prices over the links traversed from the sender. By reading a packet's `IPid` field to uniquely identify the probe type, and the TTL field to identify the LinkID, every router is able to determine whether it should modify the ECN field based on the quantized price of its outgoing link. Based on empirical data, the sequential manner in which the majority of Internet hosts increment the `IPid` of transmitted packets is conducive to observing all probe types in a relatively short block of packets with high probability. This is vital to the performance of our algorithm, because the chief source of error in estimating the path price is failing to observe one or more probe types in an

estimation block. Quantization error is the other, generally less critical source of error, and is independent of the estimation block length. We have derived the distribution of the total error, under what is the worst case scenario one could reasonably expect to see on the Internet – a host that increments the `IPid` field in a random manner. We have also derived a relatively simple upper bound on the mean-squared error under the assumption of uniformly distributed quantization error.

Our results – based on empirical data – indicate that our algorithm performs better than RAM up to certain block lengths. However, one drawback to our algorithm is the use of quantization which results in the MSE being bounded above zero regardless of block length. Thus, the MSE of RAM will eventually be lower for a sufficiently long block length, assuming that the price stays constant for the duration of the block. In future work, we will explore whether it is possible, for a given rate of price change and probing rate, to determine an optimal number of quantization bits (and hence number of unique probe types) to minimize the expected MSE.

VIII. Acknowledgments

We wish to thank NSERC – the Natural Sciences and Engineering Research Council of Canada – for sponsoring this research through their Discovery Grants Program. We also wish to thank the anonymous reviewers for their valuable comments.

References

[1] S. Athuraliya and S.H. Low, "Optimization flow control II: Implementation," Tech. Rep., Netlab, California Institute of technology, 2000.

[2] S. Athuraliya, V.H. Li, S.H. Low, and Q. Yin, "REM: Active queue management," *IEEE Network*, vol. 15, pp. 48–53, May 2001.

[3] R.J. Gibbens and F.P. Kelly, "Resource pricing and the evolution of congestion control," *Automatica*, vol. 35, pp. 1969–1985, 1999.

[4] D. Katabi, M. Handley, and C. Rohrs, "Internet congestion control for future high bandwidth-delay product environments," in *Proc. ACM Sigcomm 2002*, Pittsburgh, PA, Aug. 2002.

[5] S. Kunniyur and R. Srikant, "A time scale decomposition approach to adaptive ECN marking," in *Proc. IEEE INFOCOM*, Anchorage, AL, 2001, pp. 1330–1339.

[6] S.H. Low and D.E. Lapsley, "Optimization flow control I: Basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, pp. 861–875, Dec. 1999.

[7] F. Paganini, Z. Wang, S.H. Low, and J.C. Doyle, "A new TCP/AQM for stable operation in fast networks," in *Proc. IEEE INFOCOM*, San Francisco, CA, Apr. 2003.

[8] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion notification (ECN) to IP," Sept. 2001, IETF RFC 3168.

[9] M. Adler, J-Y Cai, J.K. Shapiro, and D. Towsley, "Estimation of congestion price using probabilitic packet marking," in *Proc. IEEE INFOCOM*, San Francisco, CA, Apr. 2003.

[10] N. Spring, D. Wetherall, and D. Ely, "Robust explicit congestion notification (ECN) signaling with nonces," June 2003, IETF RFC 3540.

[11] F. Begtasevic and P.V. Mieghen, "Measurements of the hopcount in the Internet," in *Proc. Passive and Active Measurement*, Amsterdam, The Netherlands, Apr. 2001.

[12] J. Postel, "Internet protocol," Sept. 1981, IETF RFC 791.

[13] S. Bellovin, "A technique for counting NATted hosts," in *Proc. Internet Measurement Workshop*, Marseille, France, Nov. 2002.

[14] R. Mahajan, N.T. Spring, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *Proc. ACM Sigcomm 2002*, Pittsburgh, PA, Aug. 2002.

[15] N.L. Johnson and S. Kotz, *Urn Models and their applications*, John Wiley & Sons, 1977.

[16] V.F. Kolchin, B.A. Sevastyanov, and V.P. Chistyakov, *Random allocations*, John Wiley & Sons, 1978.

[17] A. Kamath, R. Motwani, K.V. Palem, and P.G. Spirakis, "Tail bounds for occupancy and the satisfiability threshold conjecture," *Random Structures and Algorithms*, vol. 7, no. 1, pp. 59–80, 1995.

[18] D. Wetherall, D. Ely, N. Spring, S. Savage, and T. Anderson, "Robust congestion signaling," in *IEEE Conference on Network Protocols*, November 2001, pp. 332–341.