Dynamic Rate Adaptation for Improved Throughput and Delay in Wireless Network Coded Broadcast

Amy Fu, Parastoo Sadeghi and Muriel Médard

Abstract—In this paper we provide theoretical and simulation-based study of the delay performance for a number of existing throughput optimal broadcast schemes and use the results to design a new dynamic rate adaptation scheme which achieves higher transmission rates at lower packet delivery delays. Under the Bernoulli packet arrival model, the receivers' delay and throughput performance is examined. Based on their Markov states, the knowledge difference between the sender and receiver, three distinct methods for decoding are identified: zero state, leader state and coefficient-based decoding. We provide analyses of each of these and show that in most cases zero state decoding alone presents a tractable approximation of the expected packet delivery behaviour. Interestingly, while coefficient-based decoding has so far been treated as a secondary effect in the literature, we find that the choice of coefficients is extremely important in determining the delay, and a well chosen encoding scheme can in fact contribute a significant improvement to the delivery delay. Based on our delivery delay model, we develop a transmission scheme where the rate of the sender is dynamically determined using performance predictions based on feedback on the receivers' progress. We show that by utilising this feedback to control the rate, our proposed dynamic rate adaptation scheme results in significantly improved throughput and delay performance.

I. INTRODUCTION

In recent times there have been many advances in the capabilities of wireless communication. Video broadcast streaming is just one of many applications expected to grow in the future. Providing high quality service is a difficult task because there are stringent requirements in terms of delay, bandwidth and packet ordering. The challenge is therefore finding ways to utilise available resources to effectively meet these requirements [1].

In this context there are two key measures of performance. High bandwidth efficiency is essential for the transmission of higher quality videos. A measure of bandwidth efficiency is the throughput: the average number of packets successfully delivered to a receiver per unit time. But since the packets of a video can only be viewed in order, we can only consider a packet useful once it has been delivered, that is if it and all preceding packets have also been correctly received. Low delay is also desirable to avoid pauses in the video as it is being played. Therefore, it is important to minimise the delivery delay, the time between when a sender first transmits a packet to the time it is delivered to a receiver.

Meeting these requirements in a wireless setting is not an easy task [1]. Receivers’ independent channel conditions mean that they will experience very different erasure patterns, which in turn leads to a variety of demands on the sender.

A. Network coding

Linear network coding [1]–[3] is an effective way to accommodate multiple receivers’ needs while still efficiently utilising the transmission bandwidth. Under network coding the sender divides the video into equal sized packets, and combines a number of packets into each transmission using Galois field arithmetic [3]. This combination is broadcast to the receivers along with the coding coefficients used. In order to recover the original packets, receivers must collect enough packets to decode them using Gauss-Jordan elimination.

Coding coefficient selection methods can be broadly classified as either deterministic or random. Under deterministic algorithms there is a fixed procedure for determining what coefficients should be selected. Examples include [3]–[12]. By contrast, under random network coding coefficients are chosen at random, which in [13] is shown to achieve the capacity of a multicast network with probability approaching one as the size of the field becomes large. The simplicity of implementation has led to a great deal of work including [14]–[21].

Although network coding allows efficient bandwidth utilisation, the time spent waiting to receive the necessary packet combinations for decoding results in an additional decoding delay. This lower bounds the achievable delivery delay since packets can only be delivered after being decoded.

B. Methods for delay control

Generally the decoding delay incurred by network coded systems can be reduced in two ways: by limiting transmissions to a coding window and by deterministic selection of the transmitted packet combinations.

1) Coding windows: The simplest way to reduce the decoding delay is to restrict transmitted packet combinations to packets from a predetermined coding window. The idea behind this is that the smaller the coding window, the less time it will take for all packets in that window to be received and decoded. The downside, however, is that the coding window also limits the number of new packets that can be sent, which restricts the transmission rate.

Most commonly, packets are introduced in blocks [17], [18], [22], [23] or generations [19], [24]. Here the coding window is restricted to a particular block of packets, which must all be transmitted before the coding window can move on to the
next block. This is particularly common in random network coded systems, since this is the only way to restrict the delay. The advantage of block coding is that we can guarantee the decoding of all packets in the block by the time a block’s worth of innovative information is received. However the disadvantage, especially with large block sizes, is that it is also very unlikely that packets can be decoded until before the entire block is transmitted. Work including [15] highlight the effect of block size on both delay and throughput.

To obtain good delay performance while minimising negative effects on the throughput, feedback can be used to determine the most appropriate block size [22], [24]. The most appropriate block size is largely determined by the quality of feedback. In general, frequent, low delay feedback allows for finer granularity of block size control, whereas delayed or infrequent feedback must be compensated for by larger block sizes [16], [23].

However it is not always necessary to restrict the coding window to a static block. A few variants of rate control emerge for full feedback systems, where the sender is updated on the success of every packet transmission. Using feedback information it is possible to adapt the size of the coding window based on the information about receivers’ progress.

In [10], [11] a Bernoulli process governs the rate at which new packets are added to the coding window. The coding window grows as packets are added, but shrinks as packets are decoded. When a receiver has received the coding window size worth of innovative information, all packets in the coding window can be decoded. Therefore the coding window size and the delay performance depend on both the rate at which packets arrive and the receivers’ progress. This decoding behaviour is not yet well understood, and so far only asymptotic bounds on the resulting delay have been conjectured [10], [11].

In an attempt to move away from the rate restrictions of [10] and [11], [12] proposes a scheme that alternates between throughput optimality and instantaneous decodability, where the time spent in each mode is determined by the receivers’ delay performance. The corresponding effect on the coding window size is that it is allowed to grow without limitation in the first mode, but remains static in the second mode. The question then is: is there a more systematic way to select the coding window size?

2) Packet combination selection: A second form of delay minimisation is careful selection of the packet combinations transmitted. This is possible in full feedback systems, where exact knowledge of the contents of receivers’ buffers allows the sender to select combinations that increase the likelihood of receivers decoding early.

Examples of these schemes include instantly decodable network coding schemes [6]–[9], where combinations are selected to guarantee immediate decoding to a carefully-chosen subset of receivers. These rigidly guarantee low delays, but often at the sacrifice of throughput. More flexible schemes include those based on the the ‘first unseen’ packet principle [10]–[12]. These set no hard limitations on the delay, but rather attempt to improve the probability of early decoding. The mechanism of this is not yet well understood, and due to its extremely intractable nature, no analysis of this has been attempted so far in the literature. However, as we will show, by making some approximations we can gain insights into its behaviour.

C. Throughput delay tradeoffs

This tradeoff between improved throughput and increased decoding delay is a well known performance feature of network coding [12], [20], [25], [26]. In general the higher the throughput the higher the delivery delay, since at higher throughput receivers are presented with less opportunities to decode. The throughput delay tradeoff is well known, and much work [6], [7], [10]–[12], [22], [23] has been done exploring the problem of obtaining good performance given these limitations. The general approach so far has been to bound one of these parameters, then focus on improving the performance of the other. Therefore we can categorise approaches to improving the throughput delay tradeoff as follows.

On one hand we have what we will call delay limited schemes, where an upper bound is placed on the permissible delay and throughput is maximised under this constraint. These can further be categorised according to the type of delay constraint. Instantaneously decodable schemes such as [6], [7] are one example of this. Coding coefficients are selected to guarantee the immediate decodability of any new information, however this comes at the cost of lower throughput since it is often impossible to find a combination which will provide innovative information to all receivers. Hard deadline schemes also exist [22], [26], where packets must be decoded by a particular deadline, or are otherwise lost.

By contrast throughput optimal schemes such as [10], [11] work with a predetermined throughput rate, then attempt to achieve good delay performance. In [12] the rate matches that of the fastest receiver, but is offset by periods in which unencoded packets sent to reduce decoding delay. While these approaches guarantee new information for all receivers within the rate limitations, this can potentially lead to long decoding delays.

So far there has been little work on the middle ground, where a scheme is neither strictly throughput nor delay constrained. It might be asked then: is it possible to design a transmission scheme that can weigh throughput against delay, and make decisions that will simultaneously result in better throughput and delay performance?

D. Contributions

In this paper we take a first step in realising a dynamic tradeoff between delay and throughput in a wireless broadcast system. By first understanding the mechanism by which packets are delivered in Bernoulli arrival schemes such as [10], [11], we gain insight into the nature of the throughput delay tradeoff. This in itself is a difficult problem, due to the complex interactions between the sender and receivers. To manage this, we categorise the methods of delivery into three categories: zero state, leader state and coefficient-based decoding. These distinctions are made on the basis of receivers’ Markov states: defined as the difference in knowledge space between sender
and receiver at each time step. By decoupling the contributions of each method of delivery, we can present an approximation that removes the effect of cross receiver interactions. In return for some loss of accuracy, we are able to transform a mathematically intractable problem into one that gives easily-calculable results.

Based on our understanding of the mechanics of broadcast packet delivery, we propose a new transmission scheme which uses receiver feedback information to determine not only what packets should be transmitted, but also what transmission rate is most appropriate given the receivers’ immediate progress. Estimates of the sender and receivers’ rates are utilised to estimate future behaviour. This, in conjunction with complete knowledge of the receivers’ immediate progress, allows the sender to dynamically tailor the transmission rate for greatly improved performance compared with throughput optimal schemes such as [10], [11]. A related idea is considered in [22], where the coding window size is chosen to maximise the number of packets delivered to all receivers by a hard deadline. However, these restrictions are generally not conducive to good throughput, nor do they take into account the possibility of early decoding.

II. SYSTEM MODEL

Our system model is essentially the same as that in [10], [11]. A single sender aims to transmit the packets \( p_1, p_2, \ldots \) in the correct order to a set of \( R \) receivers. The sender can broadcast at the rate of at most one packet per time slot, using network coding to combine packets for transmission (see Section II-A for more details). The receivers are connected to the sender via independent erasure channels, and successfully receive these transmissions with probability \( \mu \) at each time slot.\(^1\)

Receivers store received transmissions in a buffer and send an acknowledgement after each successful reception, which we assume the sender detects without error.\(^2\) The sender uses this information to record which packets receivers have stored in their buffers.

At each time slot \( t \), the sender uses network coding to combine the packets for transmission. The packets \( p_1, p_2, \ldots \) are multiplied by coefficients \( \alpha_1(t), \alpha_2(t), \ldots \) chosen from the finite Galois field \( \mathbb{F}_M \), then summed together. This combination is transmitted along with the corresponding vector of coefficients \( v_s(t) \). If each uncoded packet \( p_i \) corresponds to the standard basis vector \( e_i \) whose \( i \)-th entry is 1, then

\[
v_s(t) = \sum_{i=1}^{\infty} \alpha_i(t)e_i = [\alpha_1(t), \alpha_2(t), \ldots]
\]

so that the \( i \)-th entry \( \alpha_i(t) \) corresponds to the coefficient of \( p_i \). Receivers use knowledge of the coding coefficients to recover the original packets by performing Gauss-Jordan elimination [3] on the packets in their buffers.

The transmitted network coded combinations are determined by two factors: the rate of packet arrivals and a decision process based on what packets the receivers have stored in their buffers. In the remainder of this section we outline the methods by which the transmitted combination is determined.

A. Arrivals

The sender restricts transmitted packet combinations to a coding window consisting of packets which have arrived in the sender queue. At each time slot \( t \) a new packet may be added to the coding window. This is called an arrival, and \( a(t) = 1 \) if a new packet is added, otherwise \( a(t) = 0 \). Once a packet has arrived, it is available to be coded in all future transmissions. If the total number of arrivals is given by

\[
A(t) = \sum_{i=0}^{t} a(i)
\]

then the sender is only permitted to code packets from the set \( p_1, \ldots, p_{A(t)} \). The transmitted combination \( c(t) \) is therefore of the form

\[
c(t) = \sum_{i=1}^{A(t)} \alpha_i(t)p_i
\]

where the coefficients \( \alpha_i(t) \) are chosen at each time slot from the field \( \mathbb{F}_M \) of an appropriate size. Schemes [10], [11] prove that it is always possible to find an innovative combination for all receivers if \( M \geq R \), the number of receivers.

Initially, we analyse a scheme where arrivals are determined by a Bernoulli process where \( \Pr(a(t) = 1) = \lambda \) independently at each time slot \( t \). By assuming the load factor \( \rho = \lambda/\mu \) is appreciably less than 1, we can provide more practical nonasymptotic analysis. Later when we expand into dynamic rate control, the arrival pattern \( a(t) \) will be determined by receiver feedback and predictive measures of throughput and decoding performance.

B. Coefficient selection

Now that the maximum size \( A(t) \) of the coding window has been established, we must choose the coding coefficients to be used at each time slot. Work up to Section VII considers schemes that maintain the innovation guarantee property, in other words all receivers who are missing packets inside the coding window are guaranteed to receive innovative information from the sender. At each time slot the sender selects coding coefficients according to one of the following algorithms.

To highlight the effects of coefficient selection on delay, we will focus on two existing schemes, the drop-when-seen transmission scheme of [10], [27] and the asymptotically optimal delivery scheme of [11], which we call schemes A and B respectively. As a means of comparison, we also consider a baseline coding scheme, which is essentially random network coding conditioned on the innovation guarantee property. The method for selecting coding coefficients in each scheme is

---

1In general receivers may have different channel erasure probabilities, but for clarity of explanation we only consider the homogeneous case.

2Although this is an impractical assumption, it greatly simplifies analysis. We will make some comments on the effect of imperfect feedback later in this work.
summarised below: more details can be found in [10], [11].

### Scheme A

Scheme A relies on the concept of seen packets. A packet $p_i$ is **seen** by a receiver if it can use the packets in its buffer to create a combination of the form $p_i + f(p_{x+1})$, where $f(p_{x+1})$ is some linear combination of the vectors $p_{x+1}, p_{x+2}, ...$. If this is not possible, then $p_i$ is **unseen**.

Scheme A ensures that with each successful reception, a receiver sees their next unseen packet. To determine what packet to send next, the sender lists the oldest unseen packet from each receiver. Moving from oldest to newest unseen packet, it adds an appropriate multiple of each packet such that it is innovative to all the corresponding receivers.

### Scheme B

Under scheme B, the sender transmits a minimal combination based on the oldest undecoded packet of each receiver. The sender lists these oldest undecoded packets and their corresponding receivers, then beginning with the newest packet in the list, it adds in older packets only if the receiver(s) that correspond to them would not otherwise receive innovative information.

### Baseline scheme

The sender transmits a random combination of all $A(t)$ packets in its current coding window. Coding coefficients are selected randomly from $\mathbb{F}_M$. For a fair comparison, feedback is used to ensure that the final packet satisfies the innovation guarantee property. If the current set of coefficients do not have this property, new random coefficients are chosen until an appropriate combination is found.

It should be noted that, unlike the baseline scheme which codes all packets in the coding window, under schemes A and B the sender will never code a packet $p_i$ until at least one of the receivers has decoded all $\{p_1, ..., p_{i-1}\}$. This is the case because schemes A and B only code the oldest unseen or undecoded packet of each user. As a result, the sender codes packets from an **effective coding window** whose size is limited by the receiver with the most packets in their buffer.

### III. Markov State

Our analysis of the delay and throughput performance of the coding schemes outlined above will be based on the receivers’ **Markov states**, a concept we will explain next. This allows us to categorise decoding methods and gives us an important tool for estimation of the delivery delay. We will define the Markov state of a receiver using the concept of virtual queue, and elaborate on how it can be used to categorise and analyse decoding behaviour.

#### A. Knowledge space and virtual queue

The **Markov state** forms the basis for our decoding analysis. It is based on the concept of **virtual queue length** which in [27] was defined as the number of packets a receiver lags behind the sender.

More formally, at time $t$ the **sender queue list** is defined as the set of vectors $V_s(t) = \{e_1, e_2, ..., e_{A(t)}\}$ corresponding to the uncoded packets $p_1, p_2, ..., p_{A(t)}$ which have arrived so far. The sender chooses packets for transmission from its **knowledge space**

$$K_s(t) = \text{span}(V_s(t)),$$

which is the set of all linear combinations the sender can compute. The size of the knowledge space is given by

$$|K_s(t)| = |V_s(t)|,$$

where $M$ is the field size and the notation $|X|$ represents the size of the set $X$. Innovative coded packets received by a receiver $r$ correspond to a set of vectors $V_r(t)$ which we call the **reception list** of receiver $r$. Likewise the knowledge space of the receiver is then given by $K_r(t) = \text{span}(V_r(t))$. The virtual queue length of a receiver can now be defined as

$$s_r(t) = |V_s(t)| - |V_r(t)|,$$

which in this paper will be referred to as the **Markov state**, for reasons explained next.

#### B. The Markov chain model

If the sender arrival rate is determined by a Bernoulli process of rate $\lambda$, then changes to a receiver’s virtual queue length over time can be modeled as a traversal through a Markov chain. This is illustrated in Fig. 1, where the states 0, 1, 2, ... correspond to the values of $s_r(t)$. For this reason we will refer to the virtual queue length $s_r(t)$ as the **Markov state** of a receiver $r$ at time $t$. Whether $s_r(t)$ increases, decreases or remains the same between time slots depends on both the arrival state at the sender $a(t)$ and the receiver’s channel conditions. The allowable state transitions for states greater than zero and their probabilities are listed in Table I. Note that although the Markov chain model is perfectly accurate for any receiver considered on its own, the fact that the sender is shared means that receivers’ Markov states can exhibit a significant amount of correlation with one another. Nevertheless this model still provides valuable insight into the delivery delay characteristics of the transmission schemes we will study.

Using the concept of Markov state we can categorise the ways in which the next packet can be delivered to a receiver

| State transition | Probability | Shorthand notation |
|------------------|-------------|--------------------|
| $s_r(t+1) = s_r(t) + 1$ | $\lambda \pi$ | $p$ |
| $s_r(t+1) = s_r(t) - 1$ | $\lambda \mu$ | $q$ |
| $s_r(t+1) = s_r(t)$ | $\lambda \mu + \lambda \pi$ | $1 - p - q$ |

**TABLE I**

The probability of transitions between Markov states for $s_r(t) > 0$, where the notation $\pi = 1 - x$ is used.

![Fig. 1. A Markov chain describing the virtual queue length.](image)
as follows.

1) Reach the zero state
A receiver \( r \) is in the zero state when its Markov state \( s_r(t) = 0 \). At this point, the number of coded packets stored at the receiver equals the number of unknown packets in the current coding window. Since schemes A, B and baseline all satisfy the innovation guarantee property, any time that \( s_r(t) = 0 \), all packets stored by the receiver are decoded.

2) Receive while a leader
Under schemes A and B, a receiver \( r \) is called a leader if it has Markov state, i.e. \( s_r(t) = \min \{ s_i(t) \} \). Since the effective window size is limited to this receiver with most packets in their buffer, receiving while a leader results in the decoding of all stored packets [27].

3) Coefficient-based decoding
Under all three schemes, even when a receiver is not leading or in the zero state, it is possible that the transmitted coded packet, combined with the combinations already stored in their buffer, allows a receiver to decode its next needed packet. Coefficient-based decoding accounts for any packets delivered at times the receiver is not either leading or in the zero state. In this case, some fraction of the stored packets are delivered.

1) Distribution of Markov states: Since the Markov state will form the basis of our analysis, the first step is to find the probability \( S_r(k) \) that at a randomly selected time, the receiver \( r \) is in state \( k \). This is equivalent to finding the stationary distribution of the Markov chain corresponding to that receiver. For the Markov chain of Fig. 1, if the arrival rate \( \lambda \) is less than the receiver channel rate \( \mu \), a stationary distribution exists such that

\[
p S_r(k) = q S_r(k + 1).
\]

Solving for \( \sum_{k=0}^{\infty} S_r(k) = 1 \), we obtain

\[
S_r(k) = \left( 1 - \frac{p}{q} \right) \left( \frac{p}{q} \right)^k.
\]

In the following Sections IV and V we will analyse the effect of Markov state on the receivers’ delivery delay.

IV. CODING WINDOW DELAY ANALYSIS
In this section we study the impact of the coding window size on the receivers’ delivery delay. We simplify this problem by dividing packet delivery mechanisms into zero state decoding, where packets are delivered only when all packets in the coding window are obtained, and leader state decoding, where packet delivery occurs when all packets in the effective coding window are received. By making this distinction we are able to provide insight into the decoding behaviour of these throughput optimal schemes that has so far been missing from the literature. Taking zero state decoding as a first approximation for our delay analysis, we use the Markov chain model to find the distribution of zero state decoding cycles, and accurately approximate the expected zero state delivery delay. Where leader state decoding has proven an intractable complication in previous analysis, we show that bounds can be found on the probability of leader state decoding, and furthermore under most circumstances it contributes only a marginal improvement to the decoding delay.

A. Zero state decoding
Here we will calculate the zero state decoding delay as an upper bound on the delivery delay of the three transmission schemes A, B and baseline. It is important to observe that as long as the innovation guarantee property holds, the Markov state of a receiver depends only on its reception rate and that of the sender. It is not affected by the coding scheme, the presence of other receivers, or even the quality of feedback. This independence makes zero state decoding analysis a valuable tool, since it allows us to make initial performance estimates without the intractable complications that have hindered the study of these transmission schemes to date.

To find the zero state decoding delay, it is not sufficient to know the proportion of time a receiver spends in the zero state, calculated in (8). The average delay largely depends on the distribution of times between returns to the zero state, which we call decoding cycles. Therefore, we will use random walk analysis to calculate the distribution of decoding cycle lengths, and based on this work, find an accurate approximation for the zero state decoding delay of Bernoulli arrival systems.

1) Decoding cycle distributions: A receiver starting in Markov state 0 experiences a decoding cycle of length \( T \) if its first return to the zero state in the Markov chain occurs after exactly \( T \) time slots. We calculate \( P_{0,0}(T) \), the probability that a decoding cycle will be of length \( T \).

We can solve this problem in two steps. First, we characterise a path through the Markov chain that consists of only moving steps where \( s_r(t+1) = s_r(t) \pm 1 \). Then we factor in the effect of pause steps, where \( s_r(t+1) = s_r(t) \).

In the first time step there are two possibilities. The receiver can remain at 0 with probability \( 1 - p \), which gives us \( P_{0,0}(1) = 1 - p \). If it instead moves up to state 1, it must return to 0 in \( T > 1 \) time steps. For a path of fixed length \( T \) to start at and return to 0, it must consist of \( 2k \) moving steps, \( k \) up and \( k \) down, and \( T - 2k \) pause steps, where \( 1 \leq k \leq \lfloor T/2 \rfloor \). If no other encounters with the zero state are permitted, the first and last time steps must be up and down steps respectively. Therefore the number of paths that first return to the 0 state in exactly \( 2k \) steps without pauses is given by the \((k-1)\)-th Catalan number [28]

\[
C_{k-1} = \frac{1}{k} \binom{2k-2}{k-1}.
\]

Now we factor in the \( T - 2k \) pauses. These pauses cannot occur in the first or last time step, otherwise the decoding cycle length would not be \( T \). For a given path of \( 2k \) moving steps, there are \( \binom{T-2}{2k-2} \) choices for pause locations. Therefore the probability of taking exactly \( T > 1 \) timeslots to return to the origin is given by

\[
P_{0,0}(T) = \sum_{k=1}^{\lfloor T/2 \rfloor} \frac{1}{k} \binom{2k-2}{k-1} \left( \frac{T-2}{2k-2} \right) p^k q^{(1-p)T-2k}.
\]
The cumulative decoding cycle length probabilities are given for a number of arrival and reception rates, $\lambda$ and $\mu$, in Fig. 2. The greater the load factor $\rho$, the more slowly the probability converges to 1 and the larger the average delay.

2) Delay approximations: Using what we know from (10) about the distribution of decoding cycle lengths, we can estimate a receiver’s average zero state decoding delay. Here this is defined as the average time between when a packet arrives at the sender to when it is decoded by the next return to zero. For a decoding cycle of length $T$ we estimate the average number of packets arriving as well as the expected delay incurred. Based on this we give an overall estimate of the decoding delay.

Where the coding cycle is of length $T = 1$, the probability that one packet arrives and is immediately decoded is simply $\lambda \mu$. This incurs no decoding delay.

In Section IV-A1 we established that, for all other coding cycles with length $T \geq 2$, the Markov state must increase in the first time slot, and decrease in the last time slot. Therefore, a packet must arrive in the first time slot, but cannot arrive in the last time slot. We now make the assumption that in the remaining $T - 2$ time slots packets arrive uniformly with probability $\lambda$. The average delay for these packets is then $T/2$. Using this approximation we can now calculate an estimate of the average zero state delivery delay.

The average number of packets delivered over a decoding cycle of length $T$ is estimated to be

$$1 + \lambda(T - 2)$$

while the delay incurred by these packets is

$$T + 0.5\lambda T(T - 2).$$

Therefore the average delay, including the $T = 1$ decoding cycle, is given by

$$\frac{\sum_{T=2}^{\infty} P_{0,0}(T)(0.5\lambda T(T - 2) + T)}{\lambda \mu + 1 + \sum_{T=2}^{\infty} P_{0,0}(T)\lambda(T - 2)}.$$  \hspace{1cm} (13)

In Fig. 3 we show how our calculated estimate, truncated at $T = 1000$, matches well with the average delivery delays obtained from simulation.

B. Leader state decoding

In this section we investigate the amount of time receivers spend leading from a nonzero state and its impact on the delivery delay, compared with zero state decoding on its own. Based on the Markov state distributions calculated in III-B1, we bound the average time that the leader(s) spend in each Markov state. Using simulations we observe the effect of leader state decoding on the delivery delay.

1) Leader state distribution: By (8) the probability of a receiver $r$ being in a state $\geq k$ is

$$S_r(\geq k) = \sum_{i=k}^{\infty} S_r(i) = \left(\frac{p}{q}\right)^k.$$  \hspace{1cm} (14)

Were receivers’ Markov states independent, the probability of having a leader in state $k$ would be

$$L(k) = \left(1 - \left(\frac{p}{q}\right)^k\right)^R.$$  \hspace{1cm} (15)

However, since the sender is common to all receivers, there is a noticeable amount of correlation between receivers’ Markov states. This is illustrated in Fig. 4, which compares the joint Markov state transition probabilities for two receivers under each model. In practice the correlated transition probabilities result in the receivers being more closely grouped together than predicted by the independent receiver model. Fig. 5 shows that the probability of leading from states $k > 0$ is higher in practice than under the independent receiver model in (15).

The probability of the leader being in state $k$ is bounded between the values in the single receiver case and the independent receiver model. Therefore, although the independent receiver model is not entirely accurate, it can still be used to make some general observations about the leader state probabilities.
need to overestimate the leader’s rate, reducing the effect of innovation guarantee property, the sender would potentially the packets the leader has received. In order to maintain the

Under imperfect feedback conditions this contribution would illustrate the impact of the number of receivers $R$.

We can observe some of these effects in Fig. 6, which

1) The probability that a receiver $r$ is leading is $\geq 1/R$, since at least one receiver must lead at each time slot.
2) Most of the time the leader will be in state $k = 0$. The larger $R$, the more likely this is the case.
3) The higher a receiver’s state $k$, the lower its likelihood of leading.
4) By (8) and (15) as the load factor $\rho \rightarrow 1^-$, or equivalently $\lambda \rightarrow \mu^-$, the state probability distribution $S_r(\geq k)$ converges on $1$ more slowly. This increases the probability that the leader will be in a higher state, and therefore the impact leader state decoding has on delay.

We can observe some of these effects in Fig. 6, which illustrates the impact of the number of receivers $R$ on the leader state decoding delay. $R = 1$ represents the extreme case where the receiver is always leading, and so results in extremely low delivery delays. As $R$ increases, however, the delay performance quickly approaches that of zero state decoding. Even at moderate values of $R$, for example $R = 10$, the contribution of leader state decoding is negligibly small. Under imperfect feedback conditions this contribution would be even smaller, since it is unknown exactly how many of the packets the leader has received. In order to maintain the innovation guarantee property, the sender would potentially need to overestimate the leader’s rate, reducing the effect of leader state decoding. As the load factor increases, however, so does the contribution from leader state decoding, consistent with observation 4.

V. COEFFICIENT-BASED DECODING

Coefficient-based decoding accounts for the remaining packets decoded while the effective coding window is incomplete. The impact of coefficient-based decoding is not well understood because of the difficulty of analysing its effects. In the literature it is generally speculated to contribute a small, if not negligible, improvement on the delivery delay. However through simulation we demonstrate two important principles for improving the likelihood of coefficient-based decoding: minimising the coding field size, and maintaining sparse codes. When these conditions are met, coefficient-based decoding can reduce the packet delivery delay significantly.

Say that at time $t$ the sender transmits a coded packet with coefficients $v_s(t)$. Then the next needed packet will be decoded only if the following condition holds.

**Lemma 1:** At time $t$, a receiver can decode the next needed packet $p_{n_e}$ iff it receives a transmission $v_s(t) \in \text{span}(K_r(t - 1) \cup e_n) \cap K_r(t - 1)$.

**Proof:** A packet $p_k$ is decoded iff $e_k \in K_r(t)$. Say that $e_n \notin K_r(t - 1)$, then for $p_n$ to be decoded at time $t$, $e_n \in \text{span}(K_r(t - 1) \cup v_s(t))$. To satisfy the innovation guarantee property, $v_s(t) \notin K_r(t - 1)$. Therefore to decode packet $p_n$ at time $t$, $v_s(t) \in \text{span}(K_r(t - 1) \cup e_n) \cap K_r(t - 1)$.

As we will show, the probability of coefficient-based decoding depends on both the transmission scheme used and the effective Markov state, which we now define.

A. Effective Markov state

The effective sender queue $V_s^e(t)$ is defined as the set of vectors corresponding to all packets the sender has transmitted in time slots $1$ to $t$. In the baseline scheme, typically $V_s^e(t) = V_s(t)$ unless all coefficients selected for the newest packet happen to be $0$. In contrast, under schemes A and B
the effective arrival rate is given by
\[ A^*_s(t) = \min \left( \max_{r \in 1, \ldots, R} \left( |V_r(t-1)| + 1, |V_s(t)| \right) \right) \]  
(16)
then \( V^*_s(t) = \{ e_1, e_2, \ldots, e_{A^*_s(t)} \} \), and similar to (4), the effective knowledge space is
\[ K^*_s(t) = \text{span}(V^*_s(t)) \].

The effective Markov state of a receiver \( r \) can then be defined as
\[ s^*_r(t) = |V^*_s(t)| - |V_r(t-1)| \].  
(17)
This differs from (6) in that, in order to calculate the probability that the current \( v_s(t) \) will be decoded, it compares the effective sender queue to the reception list prior to the current time slot.

Lemma 2: A receiver \( r \) can only coefficient-based decode its next needed packet \( p_n \) when its effective Markov state decreases, i.e. \( s^*_r(t) = s^*_r(t-1) - 1 \).

Proof: It is always true that \( K_s(t-1) \subset K^*_s(t-1) \). If the receiver is not a leader, then they have not decoded all packets in \( K^*_s(t-1) \) and \( e_n \in K^*_s(t-1) \). Therefore by Lemma 1, coefficient-based decoding can only occur if \( v_n(t) \in K^*_s(t-1) \), so that \( V^*_s(t) = V^*_s(t-1) \). Receiving an innovative packet means that \( |V_r(t)| = |V_r(t-1)| + 1 \), so by (17) \( s^*_r(t) = s^*_r(t-1) - 1 \).

So in order for coefficient-based decoding opportunity to arise, three conditions must be satisfied:
- The receiver is not a leader
- No new packets are encoded by the sender
- The receiver successfully receives the transmitted packet.

Therefore, of the time slots a receiver is neither in the zero state or a leader, approximately \( \lambda \mu \) of these provide the opportunity for coefficient-based decoding to occur. We now investigate the effectiveness of the coefficient selection schemes A and B and the baseline scheme in utilising this fraction of coefficient-based decodable time slots to minimise delay.

B. Baseline scheme

To gain some insight into the probability of coefficient-based decoding, we first study our baseline scheme. Here we will demonstrate how the effective Markov state affects the probability of coefficient-based decoding.

We first calculate for a single receiver the probability that with knowledge space \( K_r(t) \), the next needed packet \( p_n \) will be delivered. The total number of possible transmissions is given by the size of the sender’s knowledge space \( K_s(t) \) minus that of the receiver. Therefore by Lemma 1, the probability of selecting a packet under the baseline scheme which allows \( p_n \) to be decoded is
\[ \frac{|\text{span}(K_r(t-1) \cup e_n) \setminus K_r(t-1)|}{|K_s(t) \setminus K_r(t-1)|} = \frac{M-1}{M^{s^*_r(t)}-1} \]  
(18)
Therefore, the probability of coefficient-based decoding depends only on the Markov state of the receiver and the field size \( M \). The exponential dependence on both these factors means that the coefficient-based decoding probability will be very small for high effective Markov states and large field sizes.

For the multiple receiver case, simulations show that there is a negligible difference between the baseline coefficient-based decoding probabilities for the single and multiple receiver cases, provided they are coded using the same field size. Some of these probabilities, normalised over coefficient-based decodable time slots, are shown in Fig. 7, and the resulting delay performance for \( M = 4 \) is given in Fig. 8. As expected, the small coefficient-based decoding probability results in only a slight improvement over zero state decoding.

C. Scheme A

Under scheme A, the sender codes only the first unseen packet of each receiver. Furthermore the coefficient chosen is the smallest that will satisfy the innovation guarantee property. Although a field size \( M \geq R \) is necessary to guarantee innovation, the majority of the time coefficients from a much smaller field size \( F_2 \) are sufficient.

In Fig. 7 the coefficient-based decoding performance of Scheme A is compared against the baseline scheme. The performance of Scheme A with four receivers and \( F_4 \) is in fact closer to that of the baseline scheme under \( F_2 \). Since most of the time the sender effectively transmits binary field combinations, Scheme A coefficient-based decoding probabilities are only marginally worse than the \( F_2 \) baseline scheme. With 8 receivers and a field \( F_8 \), the probability of sending packets containing coefficients greater than 1 is higher. This results in decoding probabilities closer to the baseline \( F_4 \) case. In Fig. 8 we can observe that Scheme A has significantly better delay performance compared with the baseline scheme. This is primarily due to the role of leader state decoding, with a slight contribution from coefficient-based decoding.

In [10] it is suggested that any coefficient satisfying the innovation guarantee is suitable, but in our implementation the smallest allowable coefficient is chosen.
D. Scheme B

Scheme B, by contrast, attempts to closely mimic a systematic, uncoded scheme by coding additional packets into each transmission only if it is necessary to maintain the innovation guarantee. Each of these added packets has the additional property that, if received, the coded transmission will allow the corresponding receiver to decode their next needed packet, despite being a nonleading receiver.

We can expect that at least \( \lambda \) of the sender’s transmissions, corresponding to the first transmission of each new packet, will be uncoded. Fig. 9 agrees with this prediction, and we can observe that the four-receiver case has an appreciably higher proportion of uncoded packets compared with the eight receiver case. This can be attributed to the fact that the smaller the number of receivers, the lower the probability that additional packets need to be included in each transmission.

We can give an intuitive explanation for the link between sparsity and a higher probability of coefficient-based decoding. If a large fraction of undelivered packets are already decoded, this effectively reduces the size of the system of equations corresponding to unknowns in the receiver’s buffer. If the transmitted combination is itself sparse, then there is a good probability that its few nonzero elements are those previously decoded by the receiver. Where the elements corresponding to other receivers are already known, the sender will ensure that the transmitted combination allows the delivery of the receiver’s next needed packet.

Scheme B is shown in Fig. 7 to have a coefficient-based decoding probability that is significantly higher than Scheme A and decays more slowly as a function of the effective Markov state. The smaller fraction of uncoded packets means that the chance decoding probability of the eight-receiver case is somewhat less than its four-receiver counterpart. From Fig. 8 we can observe that the higher coefficient-based decoding probabilities of Scheme B result in a significantly better delivery delay compared with both Scheme A and the baseline scheme. The improvements are especially notable at high transmission rates, with almost a threefold improvement in delivery delay compared with leader state decoding.

It should be noted that under all of these transmission schemes, imperfect feedback would degrade the delivery delay performance. If the sender were to make decisions based on incomplete information about the contents of receivers’ buffers, it would be forced to transmit more conservative packet combinations, losing sparsity and relying on larger field sizes to maintain throughput optimality.

VI. Performance Predictions

In the previous two sections we analysed the decoding behaviour of the three transmission schemes: A, B and the baseline. It was found that much of the decoding behaviour depended on the Markov state of the receivers through zero, leader state and coefficient-based decoding. The delay was primarily determined by zero state decoding, with a relatively minor contribution from leader state decoding. Coefficient-based decoding was generally difficult to predict beyond certain probabilistic measures, and in any case under schemes A and baseline its effect on delay was relatively insignificant. Under imperfect feedback conditions, the contributions from leader and coefficient-based decoding would be further diminished, while zero state decoding probabilities would remain the same.

Based on these observations we will use a model of zero state decoding as an upper bound on the expected delivery delay performance. The purpose of constructing such a model is to attempt to predict the expected delay and throughput performance based on a receivers’ current Markov state \( k \). This performance is studied over a time frame of \( D \) time slots, where \( D \) gives an indication of the urgency of decoding. Two metrics \( P_d(k, D) \) and \( P_l(k, D) \) are calculated, which relate to the delivery delay and throughput, respectively. \( P_d(k, D) \) represents the average probability of zero state decoding by the deadline \( D \), while \( P_l(k, D) \) corresponds to the expected number of redundant transmissions received due to having already decoded all packets in the current coding window.
These measures will later be used to allow the sender to dynamically adjust the arrival rate and therefore the size of the coding window to give lower delay performance at higher throughputs.

A. Probability of zero state decoding

Here we calculate the probability $P_d(k, D)$ that a receiver $r$ in state $k$ will reach the zero state in the next $D$ time slots. We will approach this problem by finding the probability that the receiver first reaches 0 in exactly $D' \leq D$ time steps.

If the receiver starts at Markov state $k$, the probability it will reach the zero state in $D'$ time steps can be calculated by characterising a random walk [29] with up, down and stay probabilities of $p, q$, and $1-p-q$, respectively. To characterise this walk, we first divide its movements into $d \leq D'$ moving time steps, corresponding to changes in the walk’s position, and $D' - d$ pause steps, where the walk position does not change. Moving time steps occur with probability $p+q$, while pause steps have probability $1-p-q$. To reach the zero state the last step must be a down step, so the probability that $d$ out of $D'$ steps are moving is

$$
\binom{D' - 1}{d - 1} (1-p-q)^{D'-d}(p+q)^d.
$$

(19)

Ignoring the pause steps, we now consider a walk of length $d$ which moves up with probability $P = \frac{p}{p+q}$ and down with probability $Q = \frac{q}{p+q}$. We calculate the probability the receiver travels from $k$ to 0 for the first time in exactly $d$ moving steps. Note that this is only possible if $d = k + 2i$ for some non-negative integer $i$. If this is the case, then $u = \frac{d-k}{2}$ upwards steps and $v = \frac{d+k}{2}$ downwards steps must have been taken. We can count the number of possible paths under this constraint using the solution to the Ballot problem [29], giving

$$
N_{k,0}(d) = \binom{d}{v} \frac{k}{d}.
$$

(20)

The probability of first reaching zero in $d$ moving steps is

$$
\Pr(k \to 0, |d) = \binom{d}{v} P^u Q^v.
$$

(21)

Therefore, the probability of moving from $k$ to 0 in exactly $D'$ time steps for the first time is

$$
P_{k,0}(D') = \binom{D' - 1}{d - 1} \binom{d}{v} \frac{k}{d} (1-p-q)^{D'-d}p^u q^v.
$$

(22)

Therefore, the probability that the receiver zero state decodes by a deadline $D$ is

$$
P_d(k) = \sum_{D' = k}^D \sum_{d=k+2}^{D'} \binom{D' - 1}{d - 1} \binom{d}{v} \frac{k}{d} (1-p-q)^{D'-d}p^u q^v,
$$

(23)

where the notation in the second summation indicates that $d$ takes the value of every other integer from $k$ to $D$.

B. Number of lost transmissions

Here we find $P_l(k)$, the expected number of potential transmissions lost by a receiver $r$ in state $k$ by deadline $D$. Assuming the sender adds new packets to the coding window randomly with probability $\lambda_{est}$ at each time slot, we wish to calculate the expected number of redundant packet transmissions received by deadline $D$.

Assume that the receiver will only obtain a redundant transmission if it receives a packet while caught up to the sender. In other words, the number of time slots when a receiver’s channel is good and its Markov state is $s_r(t) = 0$. The walk’s minimum point $c$ below 0, shown in Fig. 10, corresponds to the number of packets lost over that particular walk. Again the probability it will lose $c$ packets in $D$ time steps can be calculated by characterising a random walk.

The probability of having $d$ moving steps is given by

$$
\binom{D}{d} (p+q)^d (1-p-q)^{D-d}.
$$

(24)

For a fixed starting point $k$ we need to determine the average expected minimum point, $-c$. In order to calculate this average, we individually consider walks with endpoint $b$.

A path from $k$ to $b$ of length $d$ must consist of $u = \frac{d-k+b}{2}$ upwards steps and $v = \frac{d+k-b}{2}$ downwards steps. Paths with minimum value less than or equal to $-c$ must touch $-c$ at least once. By the reflection principle [29], the number of paths from $k$ to $b$ which touch $-c$ is equal to the number of length $d$ paths from $k$ to $-2c - b$.

$$
N_{k,-2c-b}(d) = \binom{d}{(d+k+2c+b)/2}.
$$

(25)

So the probability of having a minimum state less than or equal to $-c$ is given by

$$
\Pr(k_{min} \leq -c, k \to b) = \binom{d}{(d+k+2c+b)/2} P^u Q^v.
$$

(26)

So including pause steps, this yields the probability of receiving $N_l(k) \geq c$ redundant transmissions on a walk from $k$ to $b$.

$$
\Pr(N_l(k) \geq c, k \to b) = \binom{D}{d} \binom{d}{(d+k+2c+b)/2} p^u q^v (1-p-q)^{D-d}.
$$

(27)

Summing over all values of $D$, $b$ and $c$ that could feasibly result in lost transmissions, we can find the expected number of lost transmissions:

$$
P_l(k) = \sum_c c \Pr(N_l(k) = c)
$$

$$
= \sum_{d} \sum_b \sum_c \Pr(N_l(k) \geq c, k \to b)
$$

$$
= \sum_{d=k+1}^{D} \sum_{b=k-d+2}^{\infty} \sum_{c=1}^{D} \binom{D}{d} \binom{d}{(d+k+2c+b)/2} p^u q^v (1-p-q)^{D-d}.
$$

(28)

VII. Dynamic Rate Adaptation Scheme

In the previous section we established the expected short term performance in terms of measures $P_d(k, D)$ and $P_l(k, D)$ of delay and throughput. We now use this information to define
and calculate two new metrics $M_W(r)$ and $M_X(r)$ that allow the sender to dynamically determine its transmission rate. By weighing the short term delay and throughput against each other, the sender can decide whether to wait and transmit existing packets, or send a new packet and increase the size of the coding window. The fine tuning allowed by the dynamic scheme to choose the best short term throughput delay tradeoffs allows it to achieve better long term delay and throughput performance.

### A. Decision metric

Here we outline how the sender determines whether waiting or sending is the better strategy. The sender weighs each of the receivers’ probability of zero state decoding against the expected amount of redundant information received over a time frame of length $D$.

Previously we have studied transmission schemes where the arrival pattern $a(t)$ was determined by a Bernoulli process with a fixed probability $\lambda < \mu$. On the other hand, in the dynamic rate adaptation scheme we are about to introduce, we assume that the sender has a very large (theoretically infinite) number of packets ready to be transmitted in its buffer. However, the actual transmission pattern is determined based on the receivers’ Markov states and the number of undelivered packets. Accordingly, our delay measure is adjusted to these new arrival times. It turns out that there is no straightforward way to predict the resulting arrival pattern, so we continue to model sender arrivals with a Bernoulli process, where the previously known average arrival rate $\lambda$ is replaced with an estimate $\lambda_{est}$ of the expected arrival rate. Later, we will study the impact $\lambda_{est}$ has on the throughput and delay performance.

The measure $M(r,k)$ determines how advantageous it is considered for a receiver to be in state $k$. This is based on the two previously calculated factors: $P_d(k)$, the probability of reaching the zero state in the next $D$ time slots, and $P_l(k)$ the expected number of redundant transmissions received due to catching up to the sender. $P_d(k)$ is weighted by $u$, the number of undelivered packets in the receiver’s buffer, as well as $f$, a factor which weighs the benefit of decoding existing packets against the importance of receiving new packets. The expected benefit for a particular receiver being in state $k$ is then

$$M(r,k) = u f P_d(k) - P_l(k).$$

At each time slot the sender must decide whether to wait and code only existing packets, or send and add a new packet to the coding window. It does this by calculating the expected benefit of each choice. Waiting can have two possible effects on the Markov state: it can either decrease, if the packet is successfully received, or stay the same if there is an erasure. Then by (29) the expected benefit of waiting is

$$M_W(r) = \mu u f P_d(k-1) - \mu P_l(k) + \pi u f P_d(k) - \pi P_l(k).$$

Similarly we can calculate the expected benefit of sending:

$$M_X(r) = \mu u f P_d(k) - \mu P_l(k) + \pi u f P_d(k+1) - \pi P_l(k+1).$$

Summing over all receivers, the final decision is given by the polarity of

$$\sum_{r=1}^{R} M_X(r) - \sum_{r=1}^{R} M_W(r).$$

If it is positive, then $a(t) = 1$ and sender adds a new packet to the coding window. Otherwise $a(t) = 0$ and it waits.

### B. Influences on the sender decision

To understand how the sender makes its decision, it is helpful to study the impact that a particular receiver will have, in terms of its Markov state, as well as parameters such as length of the deadline $D$ and the arrival rate estimate.

To do this, it is helpful to separate the effects of the probability of decoding $P_d(\cdot)$, and the expected number of redundant packet transmissions $P_l(\cdot)$. Since the decision depends on the polarity of $M_X(r) - M_W(r)$, we can define $P^*_d(k)$ and $P^*_l(k)$ as

$$P^*_d(k) = P_d(k) + \pi P_d(k+1) - \pi P_d(k)$$

$$P^*_l(k) = \mu P_l(k-1) + \pi P_l(k) - \mu P_l(k) - \pi P_l(k+1).$$

so that by (30) and (31), the contribution of receiver $r$ to the final decision is

$$M_X(r) - M_W(r) = u f P^*_d(k) + P^*_l(k).$$

$P^*_d(k)$ and $P^*_l(k)$ are plotted for a number of parameters in Figs. 11 and 12. When a receiver is in a particular state $k$, $P^*_l(k)$ acts as a baseline value, determining how many more packets the receiver is expected to drop on average if the sender waits a time step, compared to sending a new packet. Note that this difference can be at most one packet, since waiting once instead of sending can only increase the penalty by one packet. The effect of $P^*_d(k)$ is countered by $P^*_d(k)$, a probability whose value is scaled by the number of undelivered packets $u$ in the receiver’s queue and the weighting factor $f$.

We now examine the effect of the receivers’ Markov states, as well as the decision parameters $D$ and $\lambda_{est}$ on the sender decision.
1) Low Markov states: At $k = 0$ zero state decoding dictates that $u = 0$, so by (29) that receiver will always favour sending. Therefore, if all receivers have decoded all of the packets in the current coding window, a new packet must be added. This was not guaranteed by the Bernoulli arrival process, and is one respect in which dynamic rate control offers an improvement. At low Markov states, $P_\ast ^\dagger (k)$ indicates there is somewhat less urgency to decode since it is quite likely that the receiver will naturally decode by the deadline. However as the Markov state $k$ increases, $P_\ast ^\dagger (k)$ initially increases, representing a greater urgency for decoding, and then decreases again as the probability of decoding decreases.

2) High Markov states: At high values of $k$, we observe that receivers will make a smaller contribution to the sender decision. As $k$ becomes large, the probability of decoding by the deadline, regardless of whether the sender waits or sends, becomes small. Therefore $P_\ast (k)$ makes little contribution to the final decision. In order to lose packet transmissions, the receiver must first reach Markov state $k = 0$. So if $P_\ast (k)$ is small, then so is $P_\ast ^\dagger (k)$. Note that both $P_\ast ^\dagger (k) = P_\ast (k) = 0$ for $k > D$ since it is impossible to reach the 0 state within the $D$ time step deadline.

3) Deadline $D$: The deadline $D$ gives some sense of the urgency of decoding. In Fig. 11, the effect of decreasing $D$ is to increase the impact of $P_\ast ^\dagger (k)$ at low $k$, and reduce that of $P_\ast (k)$. This encourages the sender to favour waiting more strongly, resulting in slightly smaller Markov states and lower average delivery delay. In contrast, a larger $D$ results in a more flattened $P_\ast ^\dagger (k)$ shape and more negative $P_\ast (k)$, resulting in generally lower decoding urgency, but also a larger weighting on higher Markov states.

4) Arrival rate estimate $\lambda_{est}$: If our estimate $\lambda_{est}$ is not correct, it is important to understand the effect it will have on the decision process. As illustrated in Fig. 12, a lower estimate reduces the influence of $P_\ast ^\dagger (k)$, and shifts the peak value of $P_\ast (k)$ to lower $k$. This indicates that waiting will be favoured, even at low Markov states. At higher values of $\lambda_{est}$, decoding is considered less urgent and the sender will prefer sending until receivers are in a higher Markov state.

VIII. PERFORMANCE COMPARISON

Here we compare the performance of our dynamic rate adaptation scheme with that in the original Bernoulli arrival process model. Since, unlike the Bernoulli arrival process, the dynamic arrival scheme does not have a known arrival rate $\lambda$, we instead use throughput, the average number of arrivals, as a fair comparison. We pair the rate control scheme with the throughput optimal coding scheme $\text{B}$ of [11], since of the three schemes studied in Sections IV and V it has the best delay performance.

A. Parameters

In Fig. 13 we compare the delay throughput performance of the original random Bernoulli arrival process against that of the dynamic rate adaptation scheme, shown for different values of the decision parameters $D$, $\lambda_{est}$ and $f$. As expected, the dynamic rate adaptation scheme has much better throughput delay performance than that of the Bernoulli arrival process.

Interestingly, regardless of which parameter is being changed, the resulting performance within reasonable values appears to lie roughly on the same delay-throughput curve. The inset in Fig. 13 takes a closer look at the fine impact of decision parameters $\lambda_{est}$ and $D$ on the performance. The effect of the deadline $D$ on decision behaviour was studied in Section VII-B3. As expected, increases in $D$ increase both the throughput and delay, while low values of $D$ result in low delay, but also low throughput.

Similarly, Section VII-B4 also correctly predicted that a low $\lambda_{est}$ would result in a high throughput and delay, and vice versa. This is because the sender in effect overcompensates for the (incorrect) channel rate it expects receivers to experience in the near future. We can observe that changing $\lambda_{est}$ has a greater impact on the decision curve in Fig. 12, resulting in a larger throughput delay range compared with changing $D$ in Fig. 11.
The factor $f$, by comparison, only changes the weighting of $P_d^f(k)$ on the sender decision, without altering the shape of the decision curve. The effect is that the number of packets required to alter the decision is scaled by a factor of $1/f$. Extremely low values of $f$ reduce the influence of undelivered packets and therefore favour sending, resulting in higher throughput and higher delay. On the other hand, high values of $f$ mean that the sender will be influenced by a relatively small number of undecoded packets, so that it more likely favours waiting. Altering the factor $f$ can be used to achieve the range of desirable transmission rates, from sending whenever a receiver’s Markov state $s_r(t) < D$, if $f = 0$, to waiting whenever a receiver has an undelivered packet, if $f = \infty$.

### B. Delay distributions

Although average delay is one measure of performance, it does not give the complete picture of how long a receiver may need to wait before the packets of the video stream can be played in order. If, for example, extremely long delays occur for a small number of packets, the resulting delay performance may be quite poor.

In Fig. 14 we compare the cumulative number of packets under both Bernoulli and dynamic arrivals. Under Bernoulli arrivals, as $\lambda$ increases, so does the delay spread, with some packets experiencing delays of over 100 time slots. By comparison, the delay distribution of the dynamic rate adaptation scheme converges to 1 much more quickly under all $\lambda_{est}$ values.

### C. Rate consistency

For good performance, it is also preferable to have a consistently high packet delivery rate. As a measure of this, we find the delivery rate over a sliding window of 10 time slots. This rate distribution is plotted in Fig. 15. In general, as the throughput increases the average delivery rate also increases. However as the rate approaches $\mu$, less frequent decoding opportunities and larger decoding delays result in a less consistent transmission rate. This is particularly apparent as the throughput approaches $\mu$, where the large range of delivery rates indicates that there are periods where few or no packets are delivered, and times when a large number of packets will be delivered at once. For the Bernoulli scheme at $\lambda = 0.75$, about 9% of the time no packets will be received for all over a 10-timeslot window. The dynamic scheme at $f = 0.3$ (average throughput = 0.757) has roughly the same shape, however it has a lower probability of no deliveries over the 10-time slot window.

At lower rates, the delivery rate distribution of the dynamic scheme is more advantageous, with a much smaller variance indicating more consistent delivery throughput. There is also a much smaller proportion of time in which few or no packets are delivered. These observations both indicate that the dynamic arrival scheme can deliver packets to receivers more consistently than the Bernoulli arrival scheme.
We have demonstrated that the transmission rate of a broadcast in-order delivery scheme can be dynamically adjusted to improve both throughput and delay performance.

Analysing schemes where the transmission rate was limited by a Bernoulli arrival process, we used receivers’ Markov states to distinguish between three methods for decoding: zero state, leader state and coefficient-based decoding. We were able to accurately model the zero state decoding delay, and found that in most cases zero state decoding alone provided a reasonable approximation for the expected delivery delay. Where there were more than a few receivers, leader state decoding was observed to have a negligible impact on the delay delivery. Although the random network coding and queue management algorithm of [10] had only a small effect on the delay, using the transmission scheme of [11] resulted in significant improvements over zero state decoding alone by capitalising on more coefficient-based decoding opportunities.

Based on these observations we used a model of zero state decoding to predict the expected short term throughput and decoding behaviour. Using the results, we developed a transmission scheme that regulates the transmission rate by weighing the probability of decoding existing packets against expected losses to throughput. The throughput-delay tradeoffs under the dynamic arrival scheme resulted in much better performance, both in terms of average decoding delay and consistency of delivery rates.

So far our work has only been in the context of homogeneous networks. While our analysis is equally applicable to heterogeneous networks, a number of other issues including resource allocation and fairness must also be considered. Although our dynamic rate adaptation scheme has been shown to give improved throughput and delay, there is so far no way to predict what throughput or delay will be experienced in practice. More work needs to be done to study the relationship between decision metrics and the delay and throughput performance.

REFERENCES

[1] C. Fragouli, D. Katabi, A. Markopoulou, M. Médard, and H. Rahul, “Wireless network coding: Opportunities & challenges,” in Military Communications Conference, October 2007, pp. 1–8.

[2] R. Ahlswede, N. Cai, S. Li, and R. Yeung, “Network information flow,” IEEE Trans. Inf. Theory, vol. 46, no. 4, pp. 1204–1216, 2002.

[3] R. Koetter and M. Médard, “An algebraic approach to network coding,” IEEE/ACM Trans. Netw., vol. 11, no. 5, pp. 782–795, 2003.

[4] S. Li, R. Yeung, and N. Cai, “Linear network coding,” IEEE Trans. Inf. Theory., vol. 49, no. 2, pp. 371–381, 2003.

[5] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, “Polynomial time algorithms for multicast network code construction,” IEEE Trans. Inf. Theory., vol. 51, no. 6, pp. 1973–1982, 2005.

[6] P. Sadeghi, R. Shams, and D. Traskov, “An optimal adaptive network coding scheme for minimizing decoding delay in broadcast erasure channels,” EURASIP Journal on Wireless Communications and Networking, pp. 1–14, April 2010.

[7] S. Sorour and S. Valae, “On minimizing broadcast completion delay for instantly decodable network coding,” in Proc. IEEE International Conference on Communications (ICC), May 2010, pp. 1–5.

[8] P. Sadeghi, D. Traskov, and R. Koetter, “Adaptive network coding for broadcast channels,” in Workshop on Network Coding, Theory and Applications (NETCOD), June 2009, pp. 80–85.

[9] X. Li, C.-C. Wang, and X. Lin, “On the capacity of immediately-decodable coding schemes for wireless stored-video broadcast with hard deadline constraints,” IEEE J. Sel. Areas Commun., vol. 29, no. 5, pp. 1094–1105, May 2011.

[10] K. Sundararajan, D. Shah, and M. Médard, “ARQ for network coding,” in Proc. IEEE International Symposium on Information Theory (ISIT). IEEE, 2008, pp. 1651–1655.

[11] J. Sundararajan, P. Sadeghi, and M. Médard, “A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay,” in Workshop on Network Coding, Theory, and Applications. IEEE, 2010, pp. 1–6.

[12] J. Barros, R. Costa, D. Munaretto, and J. Widmer, “Effective delay control in online network coding,” in Proc. IEEE Conference on Computer Communications (INFOCOM), April 2009, pp. 208–216.

[13] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, “A random linear network coding approach to multicast,” IEEE Trans. Inf. Theory, vol. 52, no. 10, pp. 4413–4430, 2006.

[14] D. Lun, M. Médard, and R. Koetter, “Network coding for efficient wirelessunicast,” in Proc. International Zurich Seminar on Communications (IZS), 2006, pp. 74–77.

[15] B. Swappna, A. Eryilmaz, and N. Shroff, “Throughput-delay analysis of random linear network coding for wireless broadcasting,” in Proc. International Symposium on Network Coding (NETCOD), June 2010, pp. 1–6.

[16] D. Lucani, M. Stojanovic, and M. Médard, “Random linear network coding for time division duplexing: When to stop talking and start listening,” in Proc. IEEE Conference on Computer Communications (INFOCOM), vol. 9, 2009, pp. 1800–1808.

[17] C. Gkantsidis and P. Rodríguez, “Network coding for large scale content distribution,” in Proc. IEEE Conference on Computer Communications (INFOCOM), vol. 4, 2005.

[18] J. Park, M. Gerla, D. Lun, Y. Yi, and M. Médard, “Codecast: a network-coding-based ad hoc multicast protocol,” IEEE Wireless Commun. Mag., vol. 13, no. 5, pp. 76–81, 2006.

[19] P. Chou, Y. Wu, and K. Jain, “Practical network coding,” in Proc. Annual Allerton Conference on Communication, Control and Computing, vol. 41, no. 1. The University; 1998, 2003, pp. 40–49.

[20] A. Eryilmaz, A. Ozdaglar, M. Médard, and E. Ahmed, “On the delay and throughput gains of coding in unreliable networks,” IEEE Trans. Inf. Theory, vol. 54, no. 12, pp. 5511–5524, December 2008.

[21] M. Vehkapera and M. Médard, “A throughput-delay trade-off in packetized systems with erasures,” in Proc. IEEE International Symposium on Information Theory (ISIT). IEEE, 2005, pp. 1858–1862.

[22] L. Yang, Y. Sagduyu, J. Li, and J. Zhang. (2012) Adaptive network coding for scheduling real-time traffic with hard deadlines. [Online]. Available: http://arxiv.org/abs/1203.4008

[23] W. Zeng, C. Ng, and M. Médard. (2012) Joint coding and scheduling optimization in wireless systems with varying delay sensitivities. [Online]. Available: http://arxiv.org/abs/1202.0784

[24] C. Fragouli, D. Lun, M. Médard, and P. Pakzad, “On feedback for network coding,” in Proc. Annual Conference on Information Sciences and Systems (CISS), 2007, pp. 248–252.

[25] T. Nguyen, T. Tran, T. Nguyen, and B. Bose, “Wireless broadcast using network coding,” IEEE Trans. Veh. Technol., vol. 58, no. 2, pp. 914–925, Feb. 2009.

[26] X. Li, C.-C. Wang, and X. Lin, “Throughput and delay analysis on uncoded and coded wireless broadcast with hard deadline constraints,” in Proc. IEEE Conference on Computer Communications (INFOCOM), March 2010, pp. 1–5.

[27] J. Sundararajan, D. Shah, and M. Médard. (2009) Feedback-based online network coding. [Online]. Available: http://arxiv.org/abs/0904.1730

[28] R. Brualdi and K. Bogart, Introductory combinatorics. Prentice Hall, 1999.

[29] G. Blom, L. Holst, and D. Sandell, Polynomials. Springer-Verlag, 1994.