Beyond the Min-Cut Bound: Deterministic Network Coding for Asynchronous Multirate Broadcast

Amy Fu, Parastoo Sadeghi and Muriel Médard

Abstract

In a single hop broadcast packet erasure network, we demonstrate that it is possible to provide multirate packet delivery outside of what is given by the network min-cut. This is achieved by using a deterministic non-block-based network coding scheme, which allows us to sidestep some of the limitations put in place by the block coding model used to determine the network capacity.

Under the network coding scheme we outline, the sender is able to transmit network coded packets above the channel rate of some receivers, while ensuring that they still experience nonzero delivery rates. Interestingly, in this generalised form of asynchronous network coded broadcast, receivers are not required to obtain knowledge of all packets transmitted so far. Instead, causal feedback from the receivers about packet erasures is used by the sender to determine a network coded transmission that will allow at least one, but often multiple receivers, to deliver their next needed packet.

Although the analysis of deterministic coding schemes is generally a difficult problem, by making some approximations we are able to obtain tractable estimates of the receivers’ delivery rates, which are shown to match reasonably well with simulation. Using these estimates, we design a fairness algorithm that allocates the sender’s resources so all receivers will experience fair delivery rate performance.

I. INTRODUCTION

It is well established that the information capacity of a multicast network is given by its min-cut, a result that has been extended to broadcast packet erasure channels in [2], [3]. This is a definitive result that cannot be violated under the system model that is considered. An important feature of this model is the requirement of block coding: that is, it is assumed that a block code is used and the original information cannot be recovered until the entire block has been received. Although block coding is practical in most situations, it is not the only possible method of encoding. As a result, it is possible that the limitations imposed by block coding do not necessarily apply in general.

A related issue is the simultaneity of decoding by the receivers in the network. Block-based codes, including rateless codes (for example, those developed for degraded broadcast channels and erasure channels [4]), impose the condition that all receivers decode all information before moving onto the next block. While block-based decoding is predominantly assumed or studied, there have been some results on intermediate decoding of packets [5]–[7], albeit without considering the logical packet ordering for delivery to the application.

Linear network coding, shown in [8] to be capable of achieving network capacity, also provides some interesting counterexamples [9], [10] to the necessity of block coding restrictions. Although a large number of block-based network coding schemes exist [11]–[23] which enforce the complete reception of each block, there are also some non-block-based deterministic network coding schemes [9], [10], [24]–[28]. By relying on feedback from the receivers to determine the coded transmissions, and similar to [5]–[7], these schemes can in some cases [9], [10], [27] allow receivers to decode packets of information prior to receiving all information transmitted by the sender.

We consider the potential implications of using deterministic network coding in the case of a broadcast packet erasure network, comprising of a single sender connected to multiple receivers via a heterogeneous single-hop broadcast packet erasure channel. From [2], [29], the capacity of this network under the block coding requirement is given by the channel rate of the most challenged receiver, with the highest packet erasure probability, which is the min-cut of the network. As a consequence, even a single poorly-performing receiver can severely degrade the rate at which packets can be delivered in order to the application across the entire network. So a fundamental question that has not been considered before is:

What will happen to the delivery rates and simultaneity of decoding if the sender’s transmission rate exceeds the channel rate of one or more receivers?

Under the traditional block coding approach, the delivery rate of these receivers would be zero. Using deterministic network coding, however, it may be possible for the receivers to continue decoding new information, despite never being able to fully receive all information that has so far been transmitted by the sender. As a result, it may be possible to maintain nonzero (in-order) packet delivery rates to all receivers – even if the sender’s transmission rate exceeds their channel capacity. We demonstrate the principles behind this somewhat counterintuitive statement using the following example, which is based on the two receiver transmission scheme first presented in [24].
A. Deterministic coding example

Imagine a simple broadcast network consisting of a single sender aiming to deliver a set of packets in the correct order to two receivers: a primary receiver $Rx_1$ with channel rate $\mu_1 < 1$ and a secondary receiver $Rx_2$ with channel rate $0 < \mu_2 < \mu_1$. At each time slot $t$ the sender transmits a single packet, to which the receivers respond with an acknowledgement if they were able to receive it correctly. The sender uses this feedback to determine which coded combination should be transmitted next, as outlined in Algorithm 1. A simple example of this algorithm in progress is given in Table I. The sender’s priority is to transmit packets required by $Rx_1$, who has delivered the most packets so far, but it will also aid $Rx_2$ wherever possible.

Packet transmissions intended for $Rx_1$ are not immediately useful to $Rx_2$. However $Rx_2$ has the ability to store and make use of any of these packets it overhears. For example, in Table I at $t = 1$, $Rx_2$ stores $p_9$, which was intended for $Rx_1$. If, owing to an erasure at $Rx_1$, the sender retransmits $p_9$, it can take the opportunity to code a second packet, $p_4$, intended for $Rx_2$, into $s(t)$. This results in the time $t = 2$ transmission of $s(t) = p_9 + p_4$.

The inclusion of $p_4$ into $s(t)$ does not interfere with $Rx_1$’s ability to receive $p_9$, since $Rx_1$ has already decoded all packets that could potentially be required by $Rx_2$. Conversely, $Rx_2$ will be able to decode $p_4$ from the transmission, since it already received $p_9$ in the previous time slot. When receivers’ contributions to the transmission $s(t)$ do not impact each others’ ability to decode, something we call a knowledge differential transmission occurs. These opportunities will be discussed in a more general setting later on. The end result in this example is that the sender will, as a first priority, deliver packets to $Rx_1$, who has delivered the most packets so far, but it will also aid $Rx_2$ wherever possible.

Packet transmissions intended for $Rx_1$ are not immediately useful to $Rx_2$. However $Rx_2$ has the ability to store and make use of any of these packets it overhears. For example, in Table I at $t = 1$, $Rx_2$ stores $p_9$, which was intended for $Rx_1$. If, owing to an erasure at $Rx_1$, the sender retransmits $p_9$, it can take the opportunity to code a second packet, $p_4$, intended for $Rx_2$, into $s(t)$. This results in the time $t = 2$ transmission of $s(t) = p_9 + p_4$.

The inclusion of $p_4$ into $s(t)$ does not interfere with $Rx_1$’s ability to receive $p_9$, since $Rx_1$ has already decoded all packets that could potentially be required by $Rx_2$. Conversely, $Rx_2$ will be able to decode $p_4$ from the transmission, since it already received $p_9$ in the previous time slot. When receivers’ contributions to the transmission $s(t)$ do not impact each others’ ability to decode, something we call a knowledge differential transmission occurs. These opportunities will be discussed in a more general setting later on. The end result in this example is that the sender will, as a first priority, deliver packets to $Rx_1$, who has delivered the most packets so far, but it will also aid $Rx_2$ wherever possible.

In summary, the key idea in this paper is to break away from the block-based and rate limited schemes that are prevalent in the literature, and apply a feedback-based, deterministic network coding scheme as shown above. In doing so, we can achieve rates that exceed the limitations on capacity determined under the complete information acquisition requirement 2. In a sense, this encoding scheme relaxes the notion of broadcasting common information. Although it still ensures that all receivers can decode all packets, receivers are permitted to do so at greatly differing times. By taking advantage of the leading receiver’s progress and the stored packets in the buffers of lagging receivers, we can establish a more general form of broadcast, which we can call asynchronous broadcast, which allows receivers to deliver the same information at different rates.

Algorithm 1 Two-receiver transmission algorithm of 24.

1: Let $N_1(t)$ be the first undecoded packet of $Rx_1$.
2: if $Rx_2$ has already received $N_1(t)$ then
3: transmit the XOR combination $N_1(t) \oplus N_2(t)$.
4: else
5: transmit $N_1(t)$.
6: end if

B. Contributions and related work

In this paper we will demonstrate that it is possible for a wireless broadcast transmission scheme to achieve multi-rate packet delivery of an ordered set of information to a heterogeneous set of receivers. By using a deterministic, non-block based transmission scheme, we are able to improve upon the capacity region established in 2.

Multi-rate broadcast to heterogeneous receivers has been previously considered in the context of applications such as video broadcast and layered streaming 30, 31, where different subsets of the information are made available to different receivers. To our knowledge, the transmission scheme we will present is the first to enable the multi-rate delivery of the same set of ordered information to all receivers.

Our transmission scheme is based on a network coding scheme first proposed in 9, which aimed to minimise the in-order packet delivery delay of a homogeneous network. By making some minor modifications to this scheme, we show that it can be
adapted for transmission in a heterogeneous broadcast network. Using an analytical model to approximate the system behaviour, we demonstrate that the delivery rate of this transmission scheme can be accurately estimated. The analysis of feedback-based deterministic coding schemes is generally a challenging problem, owing to the number of cross-receiver dependencies that must be considered. We believe that our analysis is the first attempt at this level of detail.

When implementing multi-rate broadcast, one must also consider how resources are allocated amongst receivers. We therefore introduce a measure of fairness whereby a transmission scheme is considered fair if the receivers’ in-order packet delivery rates are proportional to their channel rate. To enable the sender to provide fair multi-rate delivery to the receivers, we introduce the concept of time sharing among different modes. Using a fairness algorithm to adjust the amount of time spent in each mode, we illustrate how the sender’s resources can be allocated in our transmission scheme to allow good delivery rates to be achieved by all receivers under the aforementioned fairness constraint.

II. Model

A single sender aims to transmit a set of backlogged packets \( p_1, p_2, p_3, \ldots \) in the correct order to a set of \( R \) receivers, \( RX_1, ..., RX_R \). Time is slotted, and the sender transmits at a rate of one packet per time slot. The channel between the sender and receivers is a wireless broadcast packet erasure channel, where erasures are assumed to occur according to an ergodic Bernoulli random process. More specifically, in each time slot, each receiver experiences an independent packet erasure with a probability \( \mu_i \), which is called its channel rate, as illustrated in Fig. 1 so that on average \( \mu_i \) of the broadcasted packets are successfully received. Similar erasure channel models have been considered in [2], [9], [10]. We assume that the receivers’ channel rates are unique, and order the receivers so that \( \mu_1 > \mu_2 > \ldots > \mu_R \), so that \( RX_1 \) is the fastest receiver with the highest channel rate, followed by \( RX_2, RX_3 \) and so on. For simplicity of notation, the set of channel rates is represented by the vector \( \mu = [\mu_1, \mu_2, ..., \mu_R] \). The receiver index \( i \) of a receiver \( RX_i \) is therefore a numerical ranking of its channel rate, compared to all other receivers in the system.

At each time slot \( t \) the sender broadcasts a single network coded packet \( s(t) \), along with the vector of coding coefficients used. The network coding coefficients are chosen from the Galois field \( F_M \), where the field size \( M \geq R \), typically the smallest power of 2 satisfying this requirement. For simplicity, we treat each packet \( p_n \) as a single symbol from \( F_M \).

Each receiver stores the packets it receives, along with the corresponding coding coefficients, in its own buffer. To decode the original packets, receivers use the information from the coding coefficients to apply Gaussian elimination operations, which, once enough information has been received, allow them recover the original uncoded packets. After each packet transmission, the receiver sends an acknowledgement if the packet was successfully received, or a negative acknowledgement if the packet was discarded due to an erasure. For the purposes of this work we assume the sender detects these acknowledgements without delay or error. The sender uses this information to record which packets each receiver has stored in its buffer. Based on this information, a transmission scheme is used to determine the packet combinations the sender will transmit. The components of the transmission schemes we will study are now outlined.

A. Transmission scheme

Here we give an overview of the transmission scheme we will be studying in this paper. As shown in Fig. 1 the transmission scheme consists of three components:

- A packet backlog, which contains all packets not yet delivered (to be defined shortly) to all receivers,
- A mode selector, which determines in which mode \( m \) (to be defined later) the coding scheme transmits,
- A coding scheme, which determines what network coded combination of the packets in the sender’s packet backlog will be selected for transmission.

The aim of the transmission scheme is to deliver the packets in order, though not necessarily at the same rate, to all of the receivers.

**DEFINITION 1**: A packet, \( p_n \), corresponds to the \( n^{th} \) packet in the sender’s backlog, and is represented by a symbol in \( F_M \).

**DEFINITION 2**: A transmission \( s(t) \) is a symbol in \( F_M \) that represents the network coded packet transmitted by the sender at time \( t \),

\[
s(t) = \sum_i \alpha_i(t)p_i,
\]

where the coding coefficients \( \alpha_1(t), \alpha_2, \ldots \) are symbols chosen from the field \( F_M \).

**DEFINITION 3**: A packet \( p_n \) is decoded by a receiver \( RX_i \) if Gaussian elimination on all transmissions \( s(t) \) it has received so far reveals the value of \( p_n \).

\(^2\)For our analysis, we assume there is a finite number of packets to be delivered.

\(^3\)In practice, a packet \( p_n \) would consist of a vector of \( k \) such symbols, with \( p_n \in F_M^k \), as presented in work such as [17], [25], [32], with the same coding operations applied to each element of the vector.

\(^4\)It should be noted that under our transmission scheme, it is necessary for all receivers to have enough space to store all packets being coded by the sender.
DEFINITION 4: A packet \( p_n \) is *older* than another packet \( p_m \) if its packet index \( n < m \), and *newer* if its packet index \( n \) is higher.

The packets in the sender’s backlog are ordered from oldest to newest, so packets are delivered in order from oldest to newest.

DEFINITION 5: A packet \( p_n \) is *delivered* to a receiver \( R_{x_i} \) if all older packets \( p_1, \ldots, p_n \) have been decoded by that receiver. \( d_i(t) \) is the number of packets delivered to \( R_{x_i} \) by time \( t \), so the most recently delivered packet is \( p_{d_i(t)} \).

DEFINITION 6: The *delivery rate* for \( R_{x_i} \) is given by \( R_i = d_i(t)/t \), the average rate at which packets are delivered to the receiver, as \( t \to \infty \).

Example: In Table II at \( t = 0 \), packets \( p_1, \ldots, p_3 \) have been delivered to \( R_{x_1} \), while \( p_1, \ldots, p_3 \) have been delivered to \( R_{x_2} \).

DEFINITION 7: At any time \( t \), the *next needed* packet \( N_i(t) \) of a receiver \( R_{x_i} \) is the oldest packet which has not been delivered to that receiver’s buffer. From Definition 5 \( N_i(t) = p_{d_i(t)+1} \).

Example: In Table II at time \( t = 0 \) the next needed packets of each receiver are \( N_1(0) = p_{11} \), \( N_2(0) = p_3 \) and \( N_3(0) = p_3 \).

B. Mode selection

DEFINITION 8: In *mode* \( m \), a transmission is restricted to coding packets required by receivers \( R_{x_m}, \ldots, R_{x_R} \). These receivers are said to be in the *mode* \( m \) *receivers*. A transmission made while the sender is in mode \( m \) is called a *mode* \( m \) *transmission*.

DEFINITION 9: The *mode vector* \( \beta = [\beta_1, \ldots, \beta_R] \) determines the proportion of time spent in each mode. \( \beta_m \) is the proportion of time spent in mode \( m \), and \( \sum_{m=1}^{R} \beta_m = 1 \).

At each time slot, the sender randomly chooses a mode to transmit in, selecting transmission mode \( m \) with probability \( \beta_m \) independently of previous transmission mode choices.

C. Coding scheme

The coding scheme we will study is an adaptation of the coding scheme proposed in [9]. The method for determining each transmission \( s(t) \) is given in Algorithm 2. Under this coding scheme, the sender is restricted to coding from the next needed packets of the mode \( m \) receivers, so that

\[
s(t) = \sum_{i=m}^{R} \alpha_i(t) N_i(t). \tag{2}
\]

Under Algorithm 2, the sender makes a list containing each oldest undelivered packet \( p_j \) and the corresponding set of receivers \( G_j \). Then, beginning with the newest packet in the list, it adds each packet \( p_j \) into \( s(t) \) only if the receiver(s) in \( G_j \) would otherwise receive an innovative packet.

In Algorithm 2 lines 13-15 ensure that a packet \( p_j \) is only coded into \( s(t) \) if \( 0 \in v_j \). From lines 7-8, this is contingent on at least one of the receivers in \( G_j \) being able to cancel out the existing packets in \( s(t) \) using Gaussian elimination. Therefore Algorithm 2 has the following property.

**Property 1.** In mode \( m \), only the next needed packets \( N_m(t), \ldots, N_R(t) \) of receivers \( R_{x_m}, \ldots, R_{x_R} \) can be coded into the transmission \( s(t) \). Of these, a packet \( N_i(t) \) will only be coded into \( s(t) \) if it allows \( R_{x_i} \) to deliver its next needed packet.

There are two main differences with [9]. Firstly, in this paper, rather than determining packet encodings according to a Bernoulli arrival pattern as illustrated in [9], we use modes to determine which receivers’ packets should be coded. Secondly, in [9] it is suggested that any field element may be chosen, but in our implementation, we choose the smallest coefficients that will allow innovative transmissions.

It can be shown that the *veto list* concept used in [9] only impacts the selection of the current coefficient, therefore Algorithm 2 presents a slightly simpler but equivalent form of the coding scheme in [9].

![Diagram of the components of the transmission scheme.](image_url)
Example: An example of Algorithm 2 is given in Table II for the case where each group $G_j$ consists of a single receiver. Consider $t = 2$. The coding scheme starts by setting $s(2) = p_{11}$. However, $R_{Rx_2}$ can already cancel out this transmission using $p_{11}$ in its buffer ($0 \in v_2$ in line 13 of Algorithm 2). Hence, $s(2) = p_{11} + p_3$, because $N_2(2) = p_3$. This is already innovative for $Rx_j$. Therefore, $p_3$ need not be added.

In [9] it is shown that, using this coding scheme, we can always find a transmission that is innovative for all receivers, if the field size $M > R$. It should be noted however that while every packet is innovative, receivers will still experience different delivery rates, since the packets they receive are not always contiguous, and, from Definition [5], delivery can only take place if all older packets have been decoded at that receiver.

Algorithm 2 Coding scheme.

1: Organise receivers $Rx_{m_1}, \cdots, Rx_{m_R}$ into groups $G_j$, so that $G_j$ contains all receivers $Rx_i$ whose next needed packet $N_i(t) = p_j$. Let $G$ be the set of all groups $G_j$ which contain at least one receiver.
2: Initialise $s(t) = 0$.
3: for each group $G_j \in G$, from high to low $j$, do
4: Initialise the empty veto list $v_j = \{ \}$. 
5: for each receiver $Rx_i \in G_j$ do
6: Calculate $r_i$, the result of performing Gaussian elimination on $s(t)$ with the transmissions stored in $Rx_i$’s buffer.
7: if $r_i = 0$ then
8: $v_j \leftarrow v_j \cup \{0\}$. 
9: else if $r_i = \alpha p_j$ for some field element $\alpha$ then
10: $v_j \leftarrow v_j \cup \{\alpha\}$. 
11: end if
12: end for
13: if $0 \in v_j$ then
14: $a_j \triangleq \min(\mathbb{F}_M \setminus v_j)$. 
15: Set $s(t) = s(t) + a_j p_j$. 
16: end if
17: end for

III. ANALYTICAL MODEL

It is difficult to model the coding scheme presented in Section II precisely. There is a complicated set of interdependencies between the packets transmitted by the sender and information stored in the receivers’ buffers. The sender’s transmissions are determined by the contents of the receivers’ buffers, which is in turn determined by a combination of past transmissions, the receivers’ erasure patterns and the effects of Gaussian elimination. These interdependencies makes this system difficult to model precisely. Therefore, in order to obtain a tractable model, we make some simplifying approximations in our delivery rate analysis. Here we list further approximations we will use in this paper to obtain a tractable model for calculating the receivers’ delivery rates under our transmission scheme. In Sections V to VII we will demonstrate that the numerical results obtained under this analytical model match well with simulations.

A. Simplified encoding scheme

We now analyse Algorithm 2 and show that the majority of the time the sender implements a simplified version of the coding scheme, given in Algorithm 3. Where the conditions required to use this simplified coding scheme are satisfied, we can prove that a field size $M = 2$ is sufficient to guarantee that a sender transmission $s(t)$ which is innovative for all mode $m$ receivers can always be found.

Approximation 1. For the transmission schemes studied in this paper, if the receivers’ channel rates are distinct and ordered so that $\mu_1 > \mu_2 > \cdots > \mu_R$, then their delivery rates will also obey the ordering $R_1 > R_2 > \cdots > R_R$.

Intuitively it makes sense that higher channel rates will result in higher packet delivery rates for the corresponding receivers. Where Approximation 1 holds, the total number of packets delivered by each of the receivers will diverge over time. Therefore, after a sufficient amount of time has passed the next needed packets $N_1(t), \cdots, N_R(t)$ of each receiver will also be distinct with high probability. Therefore we state the next approximation that we will use for this paper.

Approximation 2. The next needed packet of each receiver is distinct and ordered by receiver index, so that $N_1(t) > N_2(t) > \cdots > N_R(t)$.

Note however that this is not always the case: for example at $t = 0$, $N_1(t) = \cdots = N_R(t) = p_1$ since none of the receivers has received any transmissions yet.
Under Approximation 2, each receiver’s next needed packet is distinct. When this is the case, there is only a single receiver in group \( G_j \) corresponding to each next needed packet \( p_j \) and Algorithm 2 reduces to the simplified encoding scheme presented in Algorithm 3. This simplified encoding scheme, which is based on the much smaller field size \( \mathbb{F}_2 \), will form the basis of our analysis.

**Algorithm 3** Simplified encoding scheme.

1. Initialise \( s(t) = 0 \).
2. \textbf{for} \( i = 1, \ldots, R \) \textbf{do}
3. Calculate \( r_i \), the result of performing Gaussian elimination on \( s(t) \) using the packets in receiver \( Rx_i \)’s buffer.
4. \textbf{if} \( r_i = 0 \) \textbf{then}
5. \( s(t) = s(t) \oplus N_i(t) \).
6. \textbf{end if}
7. \textbf{end for}

**Property 2.** Under Algorithm 3, a mode \( m \) transmission is innovative to every mode \( m \) receiver, and non-innovative to receivers \( Rx_1, \ldots, Rx_{m-1} \).

**Proof:** Under Approximation 2, \( Rx_1 \) will have the newest next needed packet, followed by \( Rx_2, Rx_3 \), and so on. At each iteration \( i \) of the for loop, it is ensured that \( s(t) \) is innovative to \( Rx_i \). By Definition 7 and Approximation 2, \( N_{i+1}(t), \ldots, N_R(t) \) have already been decoded by \( Rx_i \), so the addition of these packets to \( s(t) \) in later iterations of the for loop does not affect the innovativeness of \( s(t) \) to \( Rx_i \). Algorithm 3 guarantees a transmission \( s(t) \) will be found which is innovative to all receivers \( Rx_m, \ldots, Rx_R \).

If the ordering of next needed packets is maintained, packets coded for \( Rx_m, \ldots, Rx_R \) are already seen by \( Rx_1, \ldots, Rx_{m-1} \). Therefore the \( m \)th mode will not contain innovative information for receivers \( Rx_1 \) to \( Rx_{m-1} \).

The simplified transmission scheme of Algorithm 3 also has the following property:

**Property 3.** Implementing Algorithm 3 with the binary field \( \mathbb{F}_2 \) is sufficient to ensure that Property 2 holds.

**Proof:** Under Approximation 2, the next needed packets of each receiver are distinct. Therefore each group \( G_j \in G \) contains exactly one receiver. It follows that in each iteration of the for loop corresponding to a group \( G_j \in G \), only one element can be added to the corresponding veto list \( v_j \). Since the binary field \( \mathbb{F}_2 \) contains two elements, this field is sufficient to ensure an appropriate coefficient can always be found.

An example of the simplified transmission scheme for three receivers in mode 1 is given in Table 1. Note that each packet coded into \( s(t) \) corresponds to a receiver being able to decode its next needed packet. Two important features of this coding scheme can be observed:

1. It is possible for packets to be delivered to any of the receivers. For example, at time \( t = 2 \), \( Rx_2 \) delivers its next needed packet \( p_5 \), and at \( t = 4 \) receivers \( Rx_1, Rx_2, Rx_3 \) all deliver their next needed packets \( p_{11}, p_7 \) and \( p_3 \), respectively.
2. Every next needed packet coded into a transmission \( s(t) \) corresponds to a receiver who can decode and hence deliver its next needed packet, if they receive the transmission. For example, at time \( t = 4 \), all three receivers’ next needed packets are encoded. Therefore, receiving the transmission allows all three receivers to deliver in the same time slot.

| \( t \) | \( s(t) \) | \( Rx_1 \) | \( Rx_2 \) | \( Rx_3 \) |
|-------|-------|-------|-------|-------|
| 0     |       | \( p_1 \cdot p_{10} \) | \( p_1 \cdot p_4, p_6 \) | \( p_1 \cdot p_{12}, p_7 \) |
| 1     | \( p_{11} \) | \( \times \) | \( p_1 \cdot p_{10} \) | \( \checkmark \) | \( p_1 \cdot p_{14}, p_6 \) | \( \times \) | \( p_1 \cdot p_7 \) |
| 2     | \( p_{11}, p_5 \) | \( \times \) | \( p_1 \cdot p_{10} \) | \( \checkmark \) | \( p_1 \cdot p_{14}, p_6 \) | \( \checkmark \) | \( p_1 \cdot p_{17}, p_{11} \) |
| 3     | \( p_{11}, p_7 \) | \( \times \) | \( p_1 \cdot p_{10} \) | \( \times \) | \( p_1 \cdot p_{14}, p_6 \) | \( \checkmark \) | \( p_1 \cdot p_{12}, p_5, p_7, p_{11} \) |
| 4     | \( p_{11}, p_7, p_3 \) | \( \checkmark \) | \( p_1 \cdot p_{11} \) | \( \checkmark \) | \( p_1 \cdot p_{17}, p_{11} \) | \( \checkmark \) | \( p_1 \cdot p_3, p_7, p_{11} \) |

**Table 1:** An example of mode 1 transmission, and the effect on receivers’ buffers. Ticks and crosses represent receivers’ channel states (successful and unsuccessful reception, respectively) at each transmission.

**B. Buffer model**

We outline the model we will use in our analysis of the receivers’ buffer’s contents. The packets stored in the buffer serve a dual purpose. Firstly, they provide information that will facilitate the decoding of the received packets. Secondly, they store information about packets that will need to be delivered at some point in the future.
For the purpose of analysis, we can imagine that every receiver’s buffer has space allocated for all of the packets in the sender’s backlog. This can be imagined as a large vector, with each element corresponding to a packet, ordered left to right from oldest to newest. The associated coding coefficients are also stored along with each element of the vector, so that the receiver has the information required to perform Gaussian elimination. As illustrated in Fig. 2, each position \( n \) in the buffer corresponds to the packet \( p_n \). The following will be used to describe the contents of a receiver’s buffer.

**DEFINITION 10.** For a receiver \( \text{Rx}_i \), the buffer coefficient matrix \( B_i \) stores the coding coefficients associated with each buffer position. Each packet \( p_n \) corresponds to the \( n \text{th} \) column of the buffer coefficient matrix, which we denote by \( B_i(n) \). If packet \( p_i \) is coded into the combination stored at \( p_n \), then the \( i \text{th} \) element of that column vector will correspond to its coding coefficient. Otherwise, this element will be 0. If buffer position \( n \) is empty, then \( B_i(n) = 0 \).

To decode the packets stored in the buffer, identical Gaussian elimination operations are applied to the columns of the buffer coefficient matrix, and the packets in the buffer. Once a column of the buffer coefficient matrix \( B_i(n) \) has been reduced to a standard basis vector \( e_n \), then the corresponding packet has been decoded.

**DEFINITION 11.** For a receiver \( \text{Rx}_i \), the buffer fill vector \( b_i \) indicates whether any information has been stored at each position \( n \) of the buffer. We say that \( b_i(n) = 0 \) if \( B_i(n) \) is the zero vector, otherwise \( b_i(n) = 1 \).

Algorithm 4 outlines the procedure for storing a received transmission \( s(t) \) in the buffer. This procedure can be divided into three stages: using Gaussian elimination to reduce \( s(t) \), storing the result in the correct buffer position, and then using this new packet to further reduce other packets in the buffer. We now give an example of Algorithm 4 at work.

**Example:** In Table II at time \( t = 2 \), \( \text{Rx}_3 \) cannot further reduce \( s(2) = p_{11} + p_5 \), nor can \( s(2) \) further reduce any of the packets stored by \( \text{Rx}_3 \). Therefore the received packet \( p_{11} + p_5 \) will be stored at buffer position \( 11 \). By contrast, at time \( t = 3 \) the presence of packets \( p_7 \) and \( p_{11} + p_5 \) allows the received packet \( s(3) = p_{11} + p_7 \) to be reduced down to \( s'(3) = p_7 \). After storing \( s'(3) = p_7 \) at buffer position \( 5 \), we use it to reduce \( p_{11} + p_5 \) down to \( p_{11} \) which will replace the content of buffer position \( 11 \).

**Algorithm 4.** Adding a received packet to the buffer of \( \text{Rx}_i \).

1. Find \( s'(t) \), the remainder when Gaussian elimination is applied to \( s(t) \) using the columns of \( B_i \).
2. if \( s'(t) \neq 0 \) then
3. Add \( s'(t) \) to buffer position \( n \), where \( n \) is the highest packet index in \( s'(t) \).
4. Use \( s'(t) \) to apply Gaussian elimination to the non-empty buffer positions, to obtain column echelon form for \( B_i \).
5. end if

**Property 4.** For every receiver \( \text{Rx}_i \), Algorithm 4 ensures that for every filled element \( n \) in the buffer, \( p_n \) is the packet with the highest index coded.

**Proof:** Algorithm 4 only modifies the contents of the buffer at lines 3 and 4. At line 3, \( s'(t) \) is stored at the correct position. Since Gaussian elimination has already been applied to \( s'(t) \), applying line 4 will not change the highest packet index of any of the stored combinations.

We now introduce some terminology that will be useful later.

**DEFINITION 12.** A packet \( p_n \) is seen to receiver \( \text{Rx}_i \) if \( b_i(n) = 1 \), i.e. there is a packet stored at buffer position \( n \).

**Example:** In Table II at time \( t = 2 \), receiver \( \text{Rx}_3 \) has packets \( p_1, p_2, p_7, p_{11}, p_5 \) in its buffer. In our model of the buffer, positions 1, 2, 7 and 11 are filled, therefore \( b_3(1) = b_3(2) = b_3(7) = b_3(11) = 1 \) and packets \( p_1, p_2, p_7, p_{11} \) have been seen. The nonzero columns of the buffer coefficient matrix are given by \( B_3(1) = e_1 \), \( B_3(2) = e_2 \), \( B_3(7) = e_7 \), and \( B_3(11) = e_{11} + e_5 \).

**DEFINITION 13.** A transmission \( s(t) \) is innovative to a receiver \( \text{Rx}_i \) if \( s(t) \) cannot be computed from packets in \( \text{Rx}_i \)’s buffer.

This method of representing the buffer has the following useful property:

**Property 5.** A receiver \( \text{Rx}_i \)’s next needed packet is always the first unseen packet in its buffer. In other words, its next needed packet is given by \( N_i(t) = p_n \), where \( n \) is the first buffer position for which \( b_i(n) = 0 \).

**Proof:** If \( b_i(1) = \cdots = b_i(n-1) = 1 \), then packets \( p_1, \cdots, p_{n-1} \) must have been decoded and delivered to the receiver (since we have received \( n-1 \) innovative combinations of \( n-1 \) packets). By Property 2 if \( p_n \) has been delivered by \( \text{Rx}_i \), then we must have \( B_i(n) = 0 \). Since \( B_i(n) = 0 \), it follows that the first unseen packet \( p_n \) is the next needed packet of \( \text{Rx}_i \).

1. **Regions of the buffer:** As illustrated in Fig. 2 our model of the buffer of a receiver \( \text{Rx}_i \) can be broken down into several distinct regions. The delivered region consists of all packets \( p_1, \cdots, p_{d_i(t)} \) which have been delivered to the receiver.

---

\( n \)In this paper we use the definition of seen introduced in [9]. Note that this differs from the definition used in [10], [25].
By definition this is immediately followed by $N_i(t) = p_{d_i(t)+1}$, the oldest undelivered packet in the buffer. Following this is the packet density region, consisting of combinations of transmitted but undelivered packets $p_{d_i(t)+2}, \ldots, N_1(t)$. In this region it is possible for $Rx_i$ to overhear both coded and uncoded transmissions intended for faster receivers. All packets lying beyond $N_1(t)$ form the empty region where no packets are yet stored, since no information about them has been broadcasted yet.

For a receiver $Rx_i$, there are three possible outcomes regarding the storage of every received packet in the buffer.

- If $s(t)$ is not innovative to $Rx_i$, then the packet will be discarded.
- If $s(t)$ is innovative and allows $N_i(t)$ to be decoded, then $s'(t)$ will be stored at position $d_i(t) + 1$. This results in the delivery of one or more new packets, so that $d_i(t + 1) > d_i(t)$.
- If $s(t)$ is innovative but it does not help decode $N_i(t)$, then the transmission is stored in the packet density region of the buffer according to Algorithm 4.

$$\text{Algorithm 4:}}$$

By definition this is immediately followed by $N_i(t) = p_{d_i(t)+1}$, the oldest undelivered packet in the buffer. Following this is the packet density region, consisting of combinations of transmitted but undelivered packets $p_{d_i(t)+2}, \ldots, N_1(t)$. In this region it is possible for $Rx_i$ to overhear both coded and uncoded transmissions intended for faster receivers. All packets lying beyond $N_1(t)$ form the empty region where no packets are yet stored, since no information about them has been broadcasted yet.

For a receiver $Rx_i$, there are three possible outcomes regarding the storage of every received packet in the buffer.

- If $s(t)$ is not innovative to $Rx_i$, then the packet will be discarded.
- If $s(t)$ is innovative and allows $N_i(t)$ to be decoded, then $s'(t)$ will be stored at position $d_i(t) + 1$. This results in the delivery of one or more new packets, so that $d_i(t + 1) > d_i(t)$.
- If $s(t)$ is innovative but it does not help decode $N_i(t)$, then the transmission is stored in the packet density region of the buffer according to Algorithm 4.

$$\text{Algorithm 4:}}$$

2) **Subregions:** Taking a closer look at the packet density region of Fig. 2 this region can be further divided into subregions 1, 2, \ldots, $R$, each with its own distinct packet density. This is illustrated in Fig. 3. The reason each of these regions exists, is that the transmissions from each mode $m$ contribute to the packet density of each subregion. However, it is not a uniform contribution. Mode 1 transmissions evenly cover packets in subregions 1, 2, \ldots, $R$, while mode 2 transmissions cover subregions 2, \ldots, $R$, and so on. The impact of these transmissions depends on how frequently they occur, which is in part determined by the mode vector $\beta$. For a given receiver $Rx_r$, the primary subregion of interest is subregion $r$, which gives us the packet density in the subregion where $Rx_r$ is delivering its next needed packets. The packet density of subregion $r$ is called the buffer density $B_r$, which will be useful in determining the delivery rate later on.

**DEFINITION 14:** For a receiver $Rx_r$, the buffer density $B_r$ is defined as the average proportion of buffer positions filled inside subregion $r$.

$$B_r = \begin{cases} 0, & r = 1 \\ \frac{\sum_{t=d_{r-1}(t)+2}^{d_{r-1}(t)+1} b_r(t)}{d_{r-1}(t)-d(t)-1}, & 1 \leq r \leq R. \end{cases} \quad (3)$$

This is illustrated in Fig. 3.

3) **Buffer model approximations:** In our delivery rate analysis of the transmission scheme, we model our system for the simple coding scheme presented in Section III-A. However, even to this coding scheme we apply some further simplification for tractability. Therefore we now present a list of approximations that will be used to model the receivers’ buffers.

**Approximation 3.** All packets stored in a receiver $Rx$’s buffer are uncoded, so if $b_1(n) = 1$, then $B_i(n) = c_n$. This means that once a receiver $Rx_i$ has seen a packet $p_n$, we assume decoding of it as well.

This approximation results a somewhat optimistic estimate of the probability of decoding. However, this is not an unreasonable approximation as our coding scheme is near-systematic, meaning that whenever Property 2 permits, the sender transmits an
uncoded packet. And of the coded transmissions received, many will be reduced to uncoded packets when they are stored in the buffer, through Gaussian elimination. From our simulations we observe that, after Gaussian elimination has been applied, the vast majority of packets stored in the buffer are uncoded. Some examples are given in Table III. In general, we observe that a few factors influence the accuracy of Approximation 5. Higher channel rates and a lower receiver index \( i \) will reduce the fraction of undecoded packets a receiver \( R_i \) has stored in its buffer. On the other hand, more time spent in modes 2, \( \cdots \), \( R \) will reduce the proportion of undecoded packets stored in the receivers’ buffers.

Example: In Table III at time \( t = 1 \), we see a typical situation where the sender sends an uncoded packet, in this case \( s(t) = p_{11} \). Although occasionally a coded packet may be stored in the receiver’s buffer, for example the buffer of \( R_3 \) at \( t = 2 \), often Gaussian elimination will help decode the packet, illustrated for \( R_3 \) at \( t = 3 \).

Approximation 5 leads to the following property, which simplifies our model of the receivers’ packet delivery mechanisms. This property will be useful for the knowledge differential calculations of Section VI.

**Property 6.** For a receiver \( R_x \) to decode its next needed packet \( N_r(t) \), every packet coded into \( s(t) \) apart from \( N_r(t) \) must have been seen by \( R_x \).

**Definition 15:** The leader in a mode \( m \) transmission is the receiver \( R_x_m \).

**Definition 16:** A leader transmission of a packet \( p_n \) is a transmission \( s(t) \) where \( p_n \) is the packet with the highest index coded into the transmission, i.e. the next needed packet of the leader. If the sender is in mode \( m \) when this transmission occurs, we can more precisely describe it as a mode \( m \) leader transmission.

Example: If the transmission \( s(t) = p_3 + p_5 + p_{10} \), then it is a leader transmission of packet \( p_{10} \), the newest packet in the transmission. If the sender was in mode 2 at time \( t \), then \( s(t) \) is a mode 2 leader transmission of \( p_{10} \).

We can further simplify our model of the receivers’ packet delivery behaviour by limiting the ways in which packets can be seen.

**Approximation 4.** A receiver \( R_x \) can only have seen an undelivered packet \( p_n \) through the reception of a leader transmission of \( p_n \).

In practice, due to the effects of Gaussian elimination, it is possible for a receiver \( R_x \) to see an undelivered packet in the packet density region at any time slot. But for the purpose of analysis, we only allow the receiver to see a packet by the most likely method, namely receiving a leader transmission. As a result of this approximation, we obtain the following property, which will also be useful in Section VI.

**Property 7.** For a packet \( p_n \) in a receiver’s packet density region to be seen, \( R_x \) must have received a leader transmission of \( p_n \).

One issue this property raises is that, since there is a common sender, the same number of leader transmissions of each packet will be sent to the receivers. As a result, we can expect some correlation between which packets each receiver has seen. These cross receiver dependencies would significantly increase the complexity of our analysis, so for the sake of simplicity we make the following approximation.

**Approximation 5.** The number of mode \( m \) leader transmissions of a packet \( p_n \) is independent of the number of mode \( b \) leader transmissions of \( p_n \), for all \( m \neq b \).

It can be observed that most of the approximations made in this section are designed to minimise the number of cross-receiver interactions we will need to take into account in our calculations. With the exception of Approximation 2 our analytical model is exact for receivers \( R_x_1 \) and \( R_x_2 \). But, as the receiver index increases, so does the impact of cross receiver interactions on the receiver’s performance. As a result, it is expected that our analytical model will lose some accuracy as the number of receivers increases. Our simulation results indicate that the delivery rates predicted by the analytical model are accurate for \( R = 4 \) and \( 8 \), however the computational complexity of the simulations prevent us from investigating higher values of \( R \) with accuracy.
IV. OUTLINE OF DELIVERY RATE ANALYSIS

In this section we outline the steps that will be taken in Sections V to VII to determine the delivery rate \( R_r \) for each receiver \( R_x_r \) under the transmission scheme outlined in Section II-C. The estimation of the delivery rate is a challenging task, given the somewhat complicated dependency between sender transmissions and the contents of each of the receivers’ buffers. The approximations made in Section III go some way toward simplifying these relationships, but the process of calculating the receivers’ delivery rates still requires some detail. A map of the quantities we will calculate on our journey towards finding the delivery rate is illustrated in Fig. 4, with definitions given in Table IV for quick reference. The values in this figure will be discussed later, but for now we simply observe that a number of dependencies exist between different components in the calculation.

To divide the problem into manageable pieces, we make the observation that there are two ways in which a packet may be delivered to a receiver. Either:

- The packet was previously seen from leader transmissions of the packet in other modes,
- The packet is delivered through a knowledge differential, while it is the receiver’s next needed packet.

The rate analysis is therefore divided into three parts. Analysis of leader transmissions takes place in Section V while knowledge differentials are studied in Section VI. The results from these two sections are combined to determine the delivery rate in Section VII. Each of these sections, corresponding to the dotted rectangles in Fig. 4, will now be briefly discussed.

A. Leader transmissions

In Section V we characterise the number of mode \( l \) leader transmissions of a packet \( p_n \) in the packet backlog. The process for calculating this probability is illustrated by the top box of Fig. 4. By Approximation 4 this characterisation is important because it can be used to determine the likelihood that a packet \( p_n \) has been seen by a receiver.

There are three possible cases to consider when finding the likelihood that a packet \( p_n \) has been transmitted \( k \) times in mode \( l \). These depend on \( R_x_l \)’s progress towards delivering \( p_n \).
| $U_l$ | Probability that a packet will not be seen from $Rx_i$ from mode 1, ..., $l$ − 1 transmissions. |
| $T_l(k)$ | Probability that there will be $k$ total mode $l$ leader transmissions of a packet, given that it was not seen from mode 1, ..., $l$ − 1 leader transmissions. |
| $\beta$ | The mode vector set by the sender (see Definition [8]). |
| $L_l(k)$ | Probability that a total of $k$ mode $l$ leader transmissions were made of a packet that has been delivered by $Rx_i$. |
| $L_l^*(k)$ | Probability that so far, there have been exactly $k$ mode $l$ leader transmissions of $N_l(t)$, $Rx_i$’s current next needed packet. |
| $\gamma_l$ | Probability of a knowledge differential transmission for $Rx_i$. |
| $D^m_n$ | Probability that the next needed packet $N_b(t)$ of a receiver $Rx_b$ is seen by another receiver $Rx_a$, where $a > b$. |
| $P_{m}(C)$ | Probability of a coding $C$ being transmitted in mode $m$. |
| $K_{m}^{l}$ | Probability that $Rx_a$ has a knowledge differential while the sender is in mode $m$. |
| $B_r$ | Expected buffer density. |
| $R_r$ | Estimated delivery rate for receiver $Rx_r$. |

**TABLE IV**

**DEFINITION OF THE VALUES USED IN FIG. 4**

- If $n \leq d_l(t)$ then $p_n$ has already been delivered by $Rx_i$, and $L_l(k)$, calculated in Section [V-D] gives the probability there were $k$ mode $l$ leader transmissions of $p_n$.
- If $n = d_l(t) + 1$, then $p_n = N_l(t)$, and $k$ takes the probability distribution $L_l^*(k)$, calculated in Section [V-C].
- If $n > d_l(t) + 1$, then there have been no mode $l$ leader transmissions of $p_n$ so far, i.e. $k = 0$, so we do not need to consider the possibility of any mode $l$ leader transmissions of these packets.

The probabilities $L_l(k)$ and $L_l^*(k)$ are calculated in Section [IV].

**B. Knowledge differentials**

In Section [VI] we find the probability of a knowledge differential. This process is illustrated by the middle box in Fig. 4. Knowledge differentials are a key concept in this paper, as they are necessary to allow receivers to deliver the remaining packets that were not seen from leader transmissions.

**DEFINITION 17**: At time $t$, a receiver $Rx_a$ has a knowledge differential if the portion of $s(t)$ coded for faster receivers $Rx_1, ..., Rx_{r-1}$ can be decoded by $Rx_a$ using the information stored in its buffer. If the sender is in mode $m$, then the probability of a knowledge differential is given by $K_{m}^{l}$. If $s(t)$ meets this criterion, it is a knowledge differential transmission. Otherwise, it is a non-knowledge differential transmission.

Knowledge differentials occur when portions of the transmission $s(t)$ intended for faster receivers do not affect $Rx_a$’s ability to decode the transmission. Knowledge differentials make it possible to provide different delivery rates to each of the receivers, since some packets will be received effectively without interference from the faster receivers. Whenever a knowledge differential transmission is received, $Rx_a$ is able to decode and hence deliver its next needed packet $N_r(t)$.

**Property 8.** A receiver $Rx_a$’s next needed packet $N_r(t)$ can only be seen through the reception of a knowledge differential transmission.

Combining Property [1] and Approximation [2] a new packet can be delivered to $Rx_a$ iff its next needed packet $N_r(t)$ is encoded into the transmission $s(t)$. In these time slots, the sender can take advantage of the knowledge differential to effectively send $Rx_a$ an uncoded packet.
There are two components required to calculate \( K'_{n}: D_{d}^{b} \), the probability that a receiver \( R_{x_{a}} \) has a \( R_{x_{b}} \) knowledge difference, and \( P_{n}(C) \), the probability of a particular coding \( C \). These are now discussed.

1) Knowledge differences: From Property 6, each of the packets coded into \( s(t) \) for other receivers must be seen by a receiver \( R_{x_{a}} \) in order to obtain a knowledge differential. The probability of each of these packets being seen is given by the probability of a \( R_{x_{b}} \) knowledge difference, defined as follows.

**Definition 18:** At time \( t \), a receiver \( R_{x_{a}} \) obtains a \( R_{x_{b}} \) knowledge difference if \( b_{a}(d_{a}(t) + 1) = 1 \), i.e. \( R_{x_{a}} \) has seen \( N_{b}(t) \). The probability of \( R_{x_{a}} \), having a \( R_{x_{b}} \) knowledge difference is \( D_{d}^{b} \). If \( a \leq b \), then we set \( D_{d}^{b} = 1 \).

This means that when \( R_{x_{a}} \) has a \( R_{x_{b}} \) knowledge difference, even if \( N_{b}(t) \) is coded in the sender transmission \( s(t) \), it will not affect \( R_{x_{a}} \)’s ability to decode \( s(t) \). Examples of this can be seen in Table II. At times \( t = 2, 3, 4 \), \( R_{x_{2}} \) has seen \( R_{x_{1}} \)’s next needed packet, \( p_{1,1} \). At time \( t = 4 \), \( R_{x_{3}} \) has seen both \( R_{x_{1}} \) and \( R_{x_{2}} \)’s next needed packets, \( p_{1,1} \) and \( p_{2,1} \), respectively.

It is important to note that knowledge differences are a common occurrence, since the broadcast nature of the system means that all transmissions may be overhead by any of the receivers. A packet seen by one receiver, but not another, causes a knowledge difference to occur in the future. Generally the higher a receiver’s channel rate, the better its chances of obtaining a knowledge difference, since they are more likely to overhear packets that are missed by other receivers.

2) Codings: The probability of a knowledge differential \( K'_{n} \) depends on which of the receivers’ next needed packets are coded into \( s(t) \). Therefore in order to determine the probability of a knowledge differential, it is first necessary to find the probability of every possible coding, \( C \).

**Definition 19:** The transmission coding \( C \) is an \( R \)-length binary vector representing which of the receivers’ next needed packets have been coded into \( s(t) \). If the \( r \)th element of \( C \), \( C(r) = 1 \), then \( N_{r}(t) \) is one of the packets coded into \( s(t) \). Otherwise, \( C(r) = 0 \) and \( s(t) \) does not code \( N_{r}(t) \).

**Example:** If \( R = 4 \) and \( s(t) = N_{1}(t) + N_{3}(t) + N_{4}(t) \), then \( C = [1, 0, 1, 1] \). If \( R = 5 \) and \( s(t) = N_{2}(t) + N_{4}(t) \), then \( C = [0, 1, 0, 1, 0] \).

### C. Delivery rate

In Section VII we will find the delivery rate \( R_{r} \). This process is illustrated in the bottom box in Fig. 4. The knowledge differential probabilities established in Section VII are used to determine the delivery rate of all of the receivers as follows.

By Property 2, every mode \( l \) transmission received by a receiver \( R_{x_{r}} \) is innovative for \( r \geq l \). Therefore, it either allows the delivery of \( N_{r}(t) \), or is stored in the packet density region. Knowledge differential transmissions allow delivery of next needed packets, and non-knowledge differential transmissions contribute seen packets to the buffer. The buffer density \( B_{r} \) (recall Definition 14) can be determined from the probability of non-knowledge differential transmissions and the delivery rates of receivers \( R_{x_{1}}, \ldots, R_{x_{r-1}} \).

For a receiver \( R_{x_{r}} \), the buffer density \( B_{r} \) gives us the fraction of packets which are already seen by \( R_{x_{r}} \), prior to the contribution of knowledge differential transmissions. The delivery rate can be found by determining how quickly knowledge differential transmissions can fill the remaining fraction of unseen packets.

### V. Leader transmissions

In this section, we find the number of mode \( l \) leader transmissions of any packet \( p_{n} \), we can expect, at any time \( t \). All calculations in this section are implicitly dependent on the mode vector \( \beta \), however for readability we omit direct mention of \( \beta \) in our notation.

#### A. Mode 1 and 2 leader transmissions

We begin with a walkthrough of the mode 1 and 2 leader transmissions of an arbitrary packet \( p_{n} \). Recall the definitions of a seen packet (Definition 12) and a leader transmission (Definition 16).

The first time a packet \( p_{n} \) will be transmitted by the sender is as a mode 1 leader transmission. This will only happen once \( p_{n} = N_{1}(t) \), i.e. it is the next needed packet of \( R_{x_{1}} \). The total number of times \( k \) that \( p_{n} \) is transmitted in mode 1 is simply the number of mode 1 transmissions it takes until it is seen by \( R_{x_{1}} \).

**Example:** Say that at time \( t = 10 \), the receiver \( R_{x_{1}} \) has delivered \( p_{1,1}, \ldots, p_{7} \), so its next needed packet is \( N_{1}(10) = p_{8} \). There are mode 1 transmissions at \( t = 10, 12, 13 \) and the corresponding \( R_{x_{1}} \) channel state at these time slots is \( \mathbf{x}, \mathbf{x}, \mathbf{v} \). In this case, at times \( t = 10, 11, 12 \), there will have been \( k = 1, 2, 3 \) mode 1 leader transmissions of \( p_{8} \), respectively, after which it is seen by \( R_{x_{1}} \). There are no more mode 1 leader transmissions of \( p_{8} \) once is seen. After this time \( p_{8} \) will become \( R_{x_{1}} \)’s next needed packet, and there will have been a total of \( k = 3 \) mode 1 leader transmissions of \( p_{8} \).

Similarly for mode 2, once \( N_{2}(t) = p_{n} \), all mode 2 transmissions will be leader transmissions of \( p_{n} \) until it is seen by \( R_{x_{2}} \). The difference from the mode 1 case is that some of the earlier mode 1 leader transmissions of a packet \( p_{n} \) may have been seen.

\[10\text{We set } D_{d}^{b} = 1 \text{ when } a \leq b, \text{ since if } a < b \text{ then } R_{x_{b}} \text{ has already decoded } N_{a}(t), \text{ and when } a = b, \text{ a receiver cannot affect its own probability of decoding.}\]
\begin{table}
\centering
\begin{tabular}{|c|c|}
\hline
\textbf{\(m\)} & \textbf{\(L_{m}(k)\)} \\
\hline
1 & \(L_{1}(k)\) \\
2 & \(L_{2}(k)\) & \(L_{2}^{*}(k)\) \\
3 & \(L_{3}(k)\) & \(L_{3}^{*}(k)\) \\
\hline
\end{tabular}
\caption{A schematic of the three-receiver probability that the sender will transmit \(k\) mode \(l\) leader transmissions of each packet \(p_{n}\).}
\end{table}

The sender only transmits mode \(l\) leader transmissions of a new packet once the previous next needed packet has been seen by \(Rx_{a}\). Recall from Approximation \(4\) that a next needed packet can be received by two methods:

- Mode \(l\) leader transmissions\(^{12}\)
- Mode \(1, \cdots, l - 1\) knowledge differential transmissions.

\(^{11}\)Modes \(b + 1, \cdots, R\) do not contribute leader transmissions because \(N_{b}(t)\) has not yet been transmitted in those modes.

\(^{12}\)Mode \(l\) transmissions are also knowledge differential transmissions, but because we are counting the number of mode \(l\) leader transmissions, we treat them separately.
The probability of a mode \(l\) transmission is simply \(\beta_l\), while the probability of a mode \(i = 1, \ldots, l - 1\) knowledge differential transmission for \(R_x_i\) is given by
\[
\gamma_l = \sum_{m=1}^{l-1} \beta_m K^m_l,
\]
where \(K^m_l\), the probability that \(R_x_i\) has a knowledge differential from a mode \(i\) transmission, will be calculated in Section VI.

So from \(R_x_i\)'s perspective, there are four possible actions from the sender at each time slot, which we now list. The sender can transmit:

1) A mode \(l\) transmission, with probability \(\beta_l\)
2) A mode \(1, \ldots, l - 1\) knowledge differential transmission, with probability \(\gamma_l\)
3) A mode \(1, \ldots, l - 1\) non-knowledge differential transmission
4) A mode \(l + 1, \ldots, R\) transmission

The probability of each outcome can be multiplied by \(\mu_l\) or \(\bar{p}_i\) to give the probability that it will or will not be received by \(R_x_i\), respectively.

Note that once the sender begins mode \(l\) leader transmissions of a new packet, it will continue to do so until it is seen, i.e. \(R_x_i\) receives either a mode \(l\) or mode \(1, \ldots, l - 1\) knowledge differential transmission of that packet. Therefore, \(L^*_l(k)\) can be considered as the probability there have been \(k\) unreceived mode \(l\) transmissions since the most recently received mode \(l\) or knowledge differential transmission.

In this context, mode \(1, \ldots, l - 1\) non-knowledge differential transmissions as well as mode \(l + 1, \ldots, R\) transmissions are not of interest, since they have no impact on the number of mode \(l\) leader transmissions. These correspond to items 3 and 4 in our list of sender actions. We can ignore these transmissions by normalising the probability of each event of interest by \(\beta_l + \mu_i\gamma_l\), the likelihood that item 1 or 2 from our list will occur.

Then the probability that an unreceived mode \(l\) transmission occurs before either a knowledge differential transmission or a received mode \(l\) transmission is given by
\[
\frac{\bar{p}_i\beta_l}{\beta_l + \mu_i\gamma_l},
\]
and the probability of a received mode \(l\) or knowledge differential transmission occurring instead of an unreceived mode \(l\) transmission is given by
\[
\frac{\mu_i\beta_l + \mu_i\gamma_l}{\beta_l + \mu_i\gamma_l}.
\]

We are now ready to determine the value of \(L^*_l(k)\). For there to have been exactly \(k\) prior mode \(l\) leader transmissions of \(R_x_i\)'s current next packet \(N_i(t)\), two requirements must be satisfied. Firstly, for there have been at least \(k\) transmissions of \(N_i(t)\), the previous \(k\) transmissions of interest must have been unreceived mode \(l\) transmissions. Secondly, to ensure that there have been no more than \(k\) transmissions of \(N_i(t)\), these unreceived mode \(l\) transmissions must have been preceded by either a received mode \(l\) or knowledge differential transmission. Combining (5) and (6), \(L^*_l(k)\) is then given by
\[
L^*_l(k) = \left(\frac{\bar{p}_i\beta_l}{\beta_l + \mu_i\gamma_l}\right)^k \mu_i\beta_l + \mu_i\gamma_l,\]
\(\beta_l + \mu_i\gamma_l\).

D. Calculating \(L_i(k)\)

In this part, we calculate the probability \(L_i(k)\) that a packet \(p_n\) that has been delivered by \(R_x_i\) was transmitted \(k\) times in total in mode \(i\). The calculation of \(L_i(k)\) depends on two separate probabilities: \(U_i\), the probability that \(p_n\) was not seen by \(R_x_i\) from mode \(1, \ldots, l - 1\) leader transmissions of \(p_n\), and \(T_i(k)\), the probability that, given \(p_n\) was unseen \(R_x_i\) from mode \(1, \ldots, l - 1\) transmissions, there were exactly \(k\) mode \(l\) leader transmissions of \(p_n\). We now calculate these values.

1) Calculating \(U_i\): We begin by calculating \(U_i\), the probability that a packet \(p_n\) will remain unseen to receiver \(R_x_i\) from mode \(1, \ldots, l - 1\) leader transmissions of the packet. The probability that \(R_x_i\) will not see \(p_n\) from mode \(i\) transmissions, where \(i < l\), is given by
\[
\sum_{k=0}^{\infty} L_i(k)\bar{p}_i^k.
\]
Under Approximation[5], we can assume that seen packets are distributed uniformly across the buffer. Therefore the probability that a packet \(p_n\) will still be unseen by \(R_x_i\) from mode \(1, \ldots, l - 1\) transmissions is given by
\[
U_i = \frac{\prod_{i=1}^{l-1} \sum_{k=0}^{\infty} L_i(k)p_i^k}{\prod_{i=1}^{l-1} \sum_{k=0}^{\infty} L_i(k)p_i^k}.
\]
2) Calculating \( T_l(k) \): Next we find \( T_l(k) \), the probability that there will be exactly \( k \) mode \( l \) leader transmissions of a packet \( p_a \), given it was unseen from mode 1, \( \ldots \), \( l-1 \) leader transmissions. As in Section V-C, only mode \( l \) transmissions and received knowledge differential transmissions impact the value of \( k \). So again, we ignore all other transmissions by normalising the probability of each relevant transmission by \( \beta_l + \mu_l \gamma_l \).

If \( k = 0 \), then a knowledge differential transmission must have been received before any mode \( l \) leader transmissions of \( p_a \) could occur. This occurs with probability

\[
T_l(0) = \frac{\mu_l \gamma_l}{\beta_l + \mu_l \gamma_l}.
\]  

(10)

For \( k > 0 \), using (5) we find that the probability of \( k - 1 \) unreceived mode \( l \) transmissions occurring prior to any receptions of knowledge differential or mode \( l \) transmissions is given by

\[
\left( \frac{\mu_l \beta_l}{\beta_l + \mu_l \gamma_l} \right)^{k-1}.
\]  

(11)

Following these \( k - 1 \) unreceived mode \( l \) transmissions, one of two things can happen to limit the sender to exactly \( k \) mode \( l \) leader transmissions of \( p_a \).

- A received mode \( l \) transmission occurs,
- An unreceived mode \( l \) transmission occurs, followed by a received knowledge differential transmission.

The probability that either of two events will occur is given by

\[
\frac{\mu_l \beta_l}{\beta_l + \mu_l \gamma_l} + \frac{\mu_l \beta_l \mu_l \gamma_l}{(\beta_l + \mu_l \gamma_l)^2}.
\]  

(12)

Combining (10), (11) and (12), the probability that exactly \( k \) mode \( l \) transmissions are required to obtain a previously unseen packet \( N_l(t) \) is given by

\[
T_l(k) = \begin{cases} 
\frac{\mu_l \gamma_l}{\beta_l + \mu_l \gamma_l}, & k = 0 \\
\frac{\mu_l \beta_l \gamma_l}{(\beta_l + \mu_l \gamma_l)^2}, & k > 0.
\end{cases}
\]  

(13)

We now scale this by \( U_l(k) \), which from (9) is the probability that \( p_a \) is unseen after mode 1, \( \cdots \), \( l-1 \) transmissions, to obtain the probability that the sender will transmit a packet \( k \) times,

\[
L_l(k) = \begin{cases} 
1 - U_l + U_l T_l(0), & k = 0 \\
T_l(k) U_l, & k > 0.
\end{cases}
\]  

(14)

E. Comparison with simulation

We examine the accuracy of our \( L_l(k) \) calculations by comparing the calculated and simulation-based probabilities in Fig. 5. We observe that calculated and simulation values match extremely well for receivers \( l = 1, 2, 3 \), but loses some accuracy for \( Rx_4 \), although it does follow the same general trend. This can be attributed to the fact our approximations lead to some loss of accuracy, particularly for higher receiver indices.

We now examine the shape of \( L_l(k) \), for different values of \( l \). For \( l = 1 \), every packet must be transmitted at least once for \( Rx_1 \) to receive it. Therefore there is no possibility of \( k = 0 \) for this receiver. The most likely case is \( k = 1 \), since \( Rx_1 \)'s high channel rate means the majority of the time, a packet will be received after one transmission. For the same reason, \( L_1(k) \) decays relatively quickly as \( k \) increases. At higher values of \( l \), however, it is reasonably likely that \( k = 0 \) because the receiver can overhear the packet from faster receivers’ leader transmissions. Although the number of leader transmissions a receiver can overhear increases as the receiver index increases, the receiver’s channel rate also decreases. In this simulation, it results in a smaller probability of \( k = 0 \) as \( l \) increases. We also observe that the distribution of \( L_l(k) \) decays more slowly for higher values of \( l \). This can also be attributed to the lower channel rates of the corresponding receivers.

VI. KNOWLEDGE DIFFERENTIALS

In this section, we will determine \( K_{m}^l \), the probability that a receiver \( Rx_r \) obtains a knowledge differential from a mode \( m \) transmission, given the mode vector \( \beta \). Through the dependencies on \( L_r^l(k) \) and \( L_l(k) \), all calculations in this section are also implicitly dependent on \( \beta \), but for readability we omit direct mention of \( \beta \) in our notation.

A. Knowledge differences

We calculate \( D_r^l \), the probability of a receiver \( Rx_a \) having a \( Rx_b \) knowledge difference. Recall from Definition 18 that if \( a \leq b \) then \( D_a^b = 1 \).
If \( b < a \), then the probability \( D_a^b \) that a receiver \( \text{Rx}_a \) has a \( \text{Rx}_b \) knowledge difference is given by the probability that \( \text{Rx}_a \) has seen \( N_b(t) \) from at least one of the mode \( 1, \cdots, b \) transmissions. This can be determined from the complementary probability of the packet \( \text{not} \) being seen from mode \( 1, \cdots, b \) transmissions, which can be calculated using (8) and (9).

\[
D_a^b = 1 - \left( \sum_{k=0}^{\infty} L_a^b(k)p_a^k \right) \prod_{l=1}^{b-1} \sum_{k=0}^{\infty} L_l(k)p_l^k.
\] (15)

Recall that in mode \( b \) transmissions, \( N_b(t) \) is the next needed packet of \( \text{Rx}_b \) and hence \( L_b^b(k) \) is used instead of \( L_l(k) \).

**B. Probability of a Coding**

We next establish the probability \( P^m(C) \) that in mode \( m \), a particular coding \( C \) will occur. The notation \( P^m(\cdot) \) represents the mode \( m \) probability of the event inside the brackets.

**DEFINITION 20:** The subcoding \( C_r \) takes the first \( r \) elements of the coding \( C \), \( C_r \) is the \( r \)-length binary vector representing which of receivers \( \text{Rx}_1 \) to \( \text{Rx}_r \)’s next needed packets have been coded into \( s(t) \).

*Example:* For the coding \( C = [1, 0, 1, 1] \), the corresponding subcodings are \( C_1 = [1] \), \( C_2 = [1, 0] \), \( C_3 = [1, 0, 1] \) and \( C_4 = [1, 0, 1, 1] \).

Combining Property 1 Property 6 and Approximation 3, a packet \( N_r(t) \) will only be coded into \( s(t) \) if \( \text{Rx}_r \) has seen all of the other packets coded into \( s(t) \). But by Approximation 2, receiver \( \text{Rx}_r \) has already decoded the next needed packets of receivers \( \text{Rx}_{r+1}, \cdots, \text{Rx}_R \). Therefore the subcoding \( C_r \) corresponds to the portion of the transmission coding \( C \) that will affect whether or not \( N_r(t) \) is coded into \( s(t) \), and therefore the value of \( C(r) \). From this information, we can break down finding the probability \( P^m(C) \) into the problem of finding the probability \( P^m(C_r) \) for \( r = 1, \cdots, R \).

In practice, it is usually not necessary to perform calculations starting from \( P^m(C_1) \). A mode \( m \) transmission will always code the next needed packet of the leader, so

\[
P^m(C(m) = 1) = 1.
\] (16)

Furthermore, since \( N_1(t), \cdots, N_{m-1}(t) \) will not be coded into a mode \( m \) transmission,

\[
C(1) = \cdots = C(m-1) = 0.
\] (17)

Combining these two pieces of information, we obtain the result,

\[
P^m(C_m) = \begin{cases} 1, & \text{for } C_m = [0, \cdots, 0, 1] \\ 0, & \text{for all other } C_m. \end{cases}
\] (18)

This means that we can start our calculations from \( C_m \), since all possible mode \( m \) subcodings begin with \([0, \cdots, 0, 1] \). Equation (18) can be used as the base case to determine the probability of the other subcodings \( C_r \), for \( r = m + 1, \cdots, R \).

Under Property 6 the only way for a packet \( N_r(t) \) to be coded is if the corresponding receiver \( \text{Rx}_r \) has seen all packets coded into the transmission \( s(t) \). Then the probability of \( N_r(t) \) being added to \( s(t) \) is the probability that \( \text{Rx}_r \) has a \( \text{Rx}_b \).
knowledge difference for all receivers \(R_{xb}\) corresponding to packets coded into \(s(t)\),

\[
P^m(C(r) = 1|C_{r-1}) = \prod_{b \in f_1^b(C_{r-1})} D_b^b \\
= \prod_{b \in f_1^b(C)} D_b^b,
\]

where for a vector \(v\), we define \(f_1^m(v)\) as the list of all indices \(r \geq m\) where \(v(r) = 1\), and \(f_0^m(v)\) as the list of all indices \(r > m\) where \(v(r) = 0\). Note that the second part of this equation is true because \(D_b^b = 1\) for all \(r \leq b\).

The probability \(P^m(C_r)\) of each subcoding \(C_r\) can then be defined in terms of the previously calculated probability \(P^m(C_{r-1})\),

\[
P^m(C(r) = 1) = P^m(C_{r-1})P^m(C(r) = 1|C_{r-1}),
\]

\[
P^m(C(r) = 0) = P^m(C_{r-1})P^m(C(r) = 0|C_{r-1}).
\]

Combining this pair of equations with (19), we can derive the probability of a particular subcoding \(C_r\):

\[
P^m(C_r) = \left( \prod_{\alpha \in f_1^m(C_r)} \prod_{b \in f_1^b(C_r)} D_b^b \right) \left( \prod_{\alpha \in f_0^m(C_r)} 1 - \prod_{b \in f_1^b(C_r)} D_b^b \right),
\]

if the first \(m\) elements of \(C_r\) are \(C_m = [0, \ldots, 0, 1]\). Otherwise, \(P^m(C_r) = 0\).

C. Knowledge differentials

We now find the probability \(K^m_r\) of a knowledge differential occurring for a receiver \(Rx_r\) under mode \(m\) transmission.

By Algorithm 3, receiver \(Rx_r\)’s next needed packet is coded iff a knowledge differential occurs. This means that we can determine the probability of a knowledge differential by finding the probability that the next needed packet of \(Rx_r\) is coded into a transmission,

\[
K^m_r = P^m(C(r) = 1) \\
= \sum_{C} C(r)P^m(C) \\
= \sum_{C_r} C_r(r)P^m(C_r).
\]

It can be noted that, in the special case where \(r = m\), we can substitute (18) to obtain

\[
K^m_m = 1.
\]

Property 10. The probability \(K^m_r\) of a mode \(m\) knowledge differential for a receiver \(Rx_r\) depends only on the ratio of \(\beta_1 : \beta_2 : \ldots : \beta_r\).

Proof: We use induction, starting with the base case of \(r = 1\). From (24) and Property 2, \(K^1_1 = 1\), and \(K^m_m = 0\) for \(m > 1\). This is true regardless of what is happening to the other receivers, so for the fastest receiver corresponding to \(r = 1\), Property (10) is satisfied.

For each subsequent value of \(r\), we examine the dependency of \(K^m_r\) on \(\beta\) using Fig. 4. We observe that dependencies on the mode vector \(\beta\) are fed in through two variables: \(T_1(k)\) and \(L_1(k)\), via \(\gamma_l\). In particular, \(K^m_r\) is calculated from \(T_1(k), \ldots, T_{r-1}(k)\) and \(L^*_1(k)\). Therefore we need only prove that \(T_1(k)\) and \(L^*_1(k)\) depend on the ratio \(\beta_1 : \beta_2 : \ldots : \beta_l\).

From (4), if \(K^1_i\) satisfies Property (10) for \(i < l\), then scaling \(\beta_1, \ldots, \beta_l\) by some factor \(\omega > 0\) will also scale \(\gamma_l\) by a factor of \(\omega\). We then make the key observation that, in each case of (13), every term in the numerator and denominator has the same multiplicity of either \(\beta_l\) or \(\gamma_l\). This means that multiplying \(\beta_1, \ldots, \beta_l\) and \(\gamma_l\) by a factor of \(\omega\) in (13) will not change the value of \(T_1(k)\). Therefore, scaling \(\beta_1, \ldots, \beta_l\) by some value \(\omega\) will not affect the value of \(L^*_1(k)\).

In (7), we similarly observe that the same multiplicity of \(\beta_l\) and \(\gamma_l\) in the numerator and denominator of \(L^*_1(k)\). Therefore both \(T_1(k)\) and \(L^*_1(k)\) depend only on the ratio \(\beta_1 : \beta_2 : \ldots : \beta_l\).

D. Simulation results

We examine the accuracy of our knowledge differential calculations. We can observe here that, for \(r = 1, 2, 3\) the calculated values of \(K^m_r\) match the simulated values extremely well, while at \(r = 4\) some inaccuracy observed at \(m = 1, 2\). This can be attributed to the cross-receiver effects from \(Rx_2\) and \(Rx_3\) in modes \(m = 1, 2\). For \(m = 3, 4\) only \(Rx_3\) and \(Rx_4\)’s packets are coded. Since only \(Rx_3\) can affect \(Rx_4\)’s knowledge differential probability, there are few cross-receiver interactions to take into account when determining \(K^m_r\), so the calculations remain accurate.
The shape of the $K^m_r$ can be explained as follows. For a given receiver $R_{x_r}$, $K^m_r$ is a relatively low probability at $m = 1$, that increases as $m$ increases. This increasing probability can be attributed to the lesser number of coded receivers, which makes it more likely that a knowledge differential will occur. At mode $m = r$, $K^m_r$ reaches its maximum value, with $K^r_r = 1$. This matches with (24). In modes $m > r$, transmissions are no longer innovative, in line with Property 2. As a result, the probability of a knowledge difference is $K^m_r = 0$.

VII. DELIVERY RATES

Now that the probability of $K^m_r$ of a mode $m$ knowledge differential is known, we are ready to estimate $R_r$, the delivery rate of each receiver $R_{x_r}$. It can be observed in Fig. 4 that $B_r$ and $R_r$ are recursively dependent on one another.

As outlined in Section III-B, we can divide all transmissions received by $R_{x_r}$ into three categories. Those that are non-innovative, those that allow the delivery of $N_r(t)$, and those that contribute to the buffer density. We use this information to estimate the delivery rate for the receivers.

A. Buffer density

We firstly determine $B_r$, receiver $R_{x_r}$’s buffer density. The first step is to determine which of the sender’s transmissions can contribute to the buffer density. From Section III-B1 a transmission must be both innovative and a non-knowledge differential transmission in order to be stored in the packet density region of the buffer. By Property 2, transmissions from modes $m = r + 1, \cdots, R$ will be noninnovative to $R_{x_r}$. From Section VI, $K^m_r$ is the fraction of the sender’s mode $m$ transmissions that will result in knowledge differentials for $R_{x_r}$ (for $m \leq r$), and from (24) $K^r_r = 1$. Therefore only the $K^m_r$ fraction of mode $m = 1, \cdots, r-1$ non-knowledge differential transmissions can contribute to $R_{x_r}$’s buffer density.

Since the distribution of leader transmissions studied in Section V is uniform, it is reasonable to say that within a subregion, the buffer density will also be uniform. Therefore, the buffer density $B_r$ is the average number of non-knowledge differential transmissions received from modes $m = 1, \cdots, r-1$ at each packet index in subregion $r$.

We consider the contribution from each mode $m < r$. For a given mode $m$, we assume all received mode $m$ non-knowledge differential packets are distributed uniformly between $p_1, \cdots, N_m(t)$. Then the average number of sender transmissions corresponding to each packet index in subregions $m, \cdots, R$ is the inverse of $R_{x_m}$’s delivery rate, $1/R_m$. Of the $\beta_m$ fraction of time spent in mode $m$, $K^m_r$ of the packets transmitted will be non-knowledge differential transmissions for $R_{x_r}$. These are distributed uniformly across $p_1, \cdots, N_m(t)$ so there is on average $\beta_m R_m$ mode $m$ transmissions per packet. Of these, $\mu_r$ will be received by $R_{x_r}$. So summing over all modes $m = 1, \cdots, r-1$,

$$B_r = \mu_r \sum_{m=1}^{r-1} \frac{\beta_m K^m_r}{R_m}. \quad (25)$$

For $r = 1$ there would be no summation, so we use this as the base case for our recursive calculation,

$$B_1 = 0. \quad (26)$$
B. Delivery rate

The task of delivering new packets to a receiver \( R_x \) is equivalently the process of making the unseen packets in the buffer seen to \( R_x \), using knowledge differential transmissions. On average, \( B_r \) of the undelivered packets in the buffer have already been seen by the receiver. Therefore it is necessary to fill in the remaining \((1 - B_r)\) fraction of packets using knowledge differential transmissions. On average, \( \mu_r \) of the knowledge differential transmissions will be received. Therefore, the delivery rate is given by

\[
R_r = \frac{\mu_r \sum_{m=1}^{r} \beta_m K^m_r}{1 - B_r}.
\]

Here we must sum the knowledge differentials contributed by each mode. Note that the summation only goes from 1 to \( r \), since from Property 2, a mode \( m \) transmission can only be innovative to receivers \( R_{x_m}, \ldots, R_{x_R} \).

We can observe from (25) and (27) that there is a recursive relationship between the delivery rates of the receivers. Note that \( B_r \) only depends on \( R_1, \ldots, R_{r-1} \) and \( K^m_r \), calculated in Section VII, while \( R_r \) only depends on \( B_r \) and \( K^m_r \). Therefore starting from the \( R_1 \) base case, where the delivery rate is \( R_1^m = \beta_1 \mu_1 \) and the buffer density is \( B_1^m = 0 \), we can sequentially calculate the delivery rates of all receivers.

C. Rate calculation algorithm

The process for calculating the delivery rate has many recursive dependencies that make it difficult to determine how to implement. Therefore, we provide Algorithm 5 to outline the order in which variables shown in Fig. 4 can be calculated.

Making use of the fact that a receiver’s delivery rate is only influenced by faster receivers, we sequentially calculate the delivery rate \( R_r \) for each receiver \( R_x \) starting from \( r = 1, \ldots, R \). Following the same structure as Sections V to VII-B we perform the leader transmission calculations, followed by knowledge differential calculations and finally the delivery rate calculations.

It is interesting to note that, in the special case of \( r = 1 \), the calculations reduce significantly. Because there are no earlier receivers for \( R_1 \)’s behaviour to be influenced by, the calculation dependencies illustrated by curved arrows in Fig. 4 can be removed.

\begin{algorithm}
  
  \textbf{Algorithm 5 Delivery rate calculation.}
  
  1: Set \( D^b_a = 1 \) for all \( a \geq b \).
  2: Set \( K^m_r = 1 \) for all \( r = m \), and \( K^m_r = 0 \) for \( r < m \).
  3: \textbf{for} \( m = 1, \ldots, R \) \textbf{do}
  4: \hspace{1em} Calculate \( \gamma_m \), using \( K^1_m, \ldots, K^{m-1}_m \).
  5: \hspace{1em} Calculate \( L^*_m(k) \), using \( \gamma_m \).
  6: \hspace{1em} Calculate \( U^*_m \) using \( L^*_1, \ldots, L^*_m(k) \).
  7: \hspace{1em} Calculate \( T^*_m(k) \) using \( \gamma_m \).
  8: \hspace{1em} Calculate \( L_m(k) \) using \( U^*_m \) and \( T^*_m(k) \).
  9: \hspace{1em} \textbf{for} \( a = m + 1, \ldots, R \) \textbf{do}
  10: \hspace{2em} Calculate \( D^m_a \) using \( L^*_1(k), \ldots, L^*_m(k) \) and \( L^*_m(k) \).
  11: \hspace{1em} \textbf{end for}
  12: \hspace{1em} \textbf{end for}
  13: \hspace{1em} \textbf{for} \( C^m_m \) \textbf{do}
  14: \hspace{2em} Calculate \( P^a_n(C_m) \) using \( D^b_a \) for \( b < a < m \).
  15: \hspace{1em} \textbf{end for}
  16: \hspace{1em} \textbf{for} \( r = n, \ldots, m \) \textbf{do}
  17: \hspace{2em} Calculate \( K^b_m \) using all \( P^a_n(C_r) \).
  18: \hspace{1em} \textbf{end for}
  19: \hspace{1em} \textbf{end for}
  20: \hspace{1em} Calculate \( B^*_m \) using \( K^1_m, \ldots, K^{m-1}_m \) and \( R_1, \ldots, R_{m-1} \).
  21: \hspace{1em} Calculate \( R_m \) using \( K^1_m, \ldots, K^{m}_m \) and \( B_m \).
  22: \textbf{end for}
  
\end{algorithm}

D. Simulation results

We now compare the simulated and calculated delivery rates for the receivers. Fig. 7 illustrates the simulated and calculated delivery rates of a set of receivers under the settings listed in Table VI.

\[\text{In our calculations, empty summations are equal to zero, and empty products are equal to one.}\]
Fig. 7. Probability of a knowledge differential for a receiver $Rx_r$ in mode $m$, for the simulation settings given in Table VI.

| Setting | $\mu$ | $\beta$ |
|---------|--------|---------|
| A       | $[0.8, 0.6, 0.4, 0.2]$ | $[0.85, 0.05, 0.05, 0.05]$ |
| B       | $[0.9, 0.8, 0.7, 0.6, 0.5]$ | $[0.8, 0.05, \ldots, 0.05]$ |
| C       | $[0.8, 0.75, 0.7, \ldots, 0.45]$ | $[1.0, \ldots, 0]$ |
| D       | $[0.8, 0.75, \ldots, 0.45]$ | $[0.86, 0.02, \ldots, 0.02]$ |

TABLE VI SIMULATION CONDITIONS USED IN Fig. 7

Under Settings A and B, the calculated delivery rates match well with the values observed in simulation. In settings C and D on the other hand, the approximate calculations are fairly accurate for the first few receivers, but gradually lose accuracy for higher receiver indices. The calculated values tend to be a little optimistic as the receiver index grows, since at higher receiver indices, Approximation 3 loses its accuracy. Setting C represents the extreme case, where all sender transmissions are allocated to mode 1. For a receiver $Rx_r$, the closer the mode $m$ is to $r$, the more accurate the knowledge differential estimate will be. Therefore greater the amount of time allocated to higher modes, the more accurate the calculated rates will be.

It is interesting to note that, for Setting A, despite some slight inaccuracies in the knowledge differential estimates of Fig. 6, the delivery rates under both simulation and calculation match quite well. In part, this can be attributed to the inverse relationship between the buffer density and the probability of knowledge differential that can be observed in (25). As a result of this relationship, inaccuracies in the calculated probability of a knowledge differential are partially mitigated by their inverse effect on the buffer density.

VIII. FAIRNESS

We now use the delivery rate analysis of Sections VI and VII to determine the value of the mode vector $\beta$ that will result in fair delivery rates for all receivers. We define fairness as follows:

**Definition 21**: A transmission scheme is *fair* for all receivers if their delivery ratios are equal, so that

$$\frac{R_1}{\mu_1} = \frac{R_2}{\mu_2} = \cdots = \frac{R_R}{\mu_R}.$$  \hspace{1cm} (28)

Our aim then is to achieve equal *delivery ratios* for all receivers, where the delivery ratio $Q_r$ of a receiver $Rx_r$ is defined as

$$Q_r = \frac{R_r}{\mu_r}.$$  \hspace{1cm} (29)

In practice, the method we will outline can be modified to equalise any set of delivery ratios, so long as doing so will

\[ Note, however, that $\beta$ must be restricted to values that maintain the ordering of the receivers under Approximation 2. \]
maintain the condition
\[ R_1 \geq R_2 \geq \cdots \geq R_R. \] (30)

A. The rate scaling property

Before we can begin, it is necessary to highlight the key property of our transmission scheme that will help us build our fairness algorithm.

**Property 11.** Scaling \( \beta_1, \cdots, \beta_r \) by some amount \( \omega > 0 \) does not affect the buffer density of receivers \( R_{x_1}, \cdots, R_{x_r} \), and scales their delivery rates by \( \omega \).

**Proof:** We will prove this property using induction. Let us substitute \( \beta'_m = \omega \beta_m \) for all \( m = 1, \cdots, r \) into (27) and (25). Using the notation \( B_r(\beta) \) and \( R_r(\beta) \) to show the explicit dependency on the mode vector \( \beta \), we obtain
\[
B_r(\beta') = \mu_r \sum_{m=1}^{r-1} \omega \beta_m K^m_r R_m(\beta') \] (31)
\[
R_r(\beta') = \frac{\sum_{m=1}^{r} \omega \beta_m K^m_r}{1 - B_r(\beta')} \] (32)

As the base case, \( R_1(\beta') = \omega \beta_1 \mu_1 \) and \( B_1(\beta') = 0 \). \( R_1(\beta') \) is linearly dependent on \( \omega \), and \( B_1(\beta') \) is a constant. Therefore Property [11] holds for the base case.

We now show that, if Property [11] is true for \( R_{x_1}, \cdots, R_{x_{r-1}} \), then it also holds for \( R_{x_r} \). From Property [10] we know that scaling \( \beta_1, \cdots, \beta_r \) by \( \omega \) will not affect the probability \( K^m_r \) of a knowledge differential. Now, assume that \( R_m(\beta') = \omega R_m(\beta) \) and \( B_m(\beta') = B_m(\beta) \) for \( m < r \). Then by substituting these values into (32) and (31), we obtain
\[
B_r(\beta') = \mu_r \sum_{m=1}^{r-1} \omega \beta_m K^m_r R_m(\beta) \]
\[
R_r(\beta') = \frac{\sum_{m=1}^{r} \omega \beta_m K^m_r}{1 - B_r(\beta)} \] (33)

and
\[
R_r(\beta') = \frac{\sum_{m=1}^{r} \omega \beta_m K^m_r}{1 - B_r(\beta)} = \omega R_r(\beta). \] (34)

Therefore, Property [11] holds for all receivers.

This is a key result since it tells us that, for a receiver \( R_{x_r} \), scaling the time spent in modes \( 1, \cdots, r \) by a factor of \( \omega \) will scale the delivery rates \( R_1, \cdots, R_r \) by a factor of \( \omega \) as well. This will give us a predictable way to determine the receivers’ delivery rates as we adjust the value of the mode vector \( \beta \).

B. Fairness algorithm

We now outline the fairness algorithm used to determine the value of \( \beta \) required to achieve delivery rate fairness. Recall that, in Section [VII] we calculated both the expected buffer density and delivery rates, given \( \beta \). The aim now is to determine the value of \( \beta \) that will result in fair delivery rates for all receivers.

The receivers’ delivery ratios can be matched using the procedure outlined in Algorithm [6]. This fairness algorithm performs two functions as it iterates through each value of \( r \). Firstly, it scales down the amount of time allocated to modes \( 1, \cdots, r-1 \) by the calculated quantity \( 1 - \omega[r] \). Secondly, it allocates the remaining \( \omega[r] \) fraction of time to mode \( r \). By Properties [2] and [11] neither of these actions affects the delivery ratio of receivers \( 1, \cdots, r-1 \). Therefore, Algorithm [6] ensures that equalised delivery ratios will remain matched through subsequent iterations of the algorithm. After the \( r \)th iteration is complete, \( \beta[r] \) will allow the sender to achieve \( Q_1[r] = \cdots = Q_r[r] \).

1) Notation: To reflect the changing values at each iteration, we append the variables with an index \( [r] \) to represent the updated values as of the \( r \)th iteration. \( \beta_m[r] \) represents the time allocated to mode \( m \) after iteration \( r \), and \( \omega[r] \) is an equalisation factor which determines \( \beta[r] \) based on \( \beta[r-1] \), the value at the previous iteration. The delivery ratio \( Q_m[r] \) of each receiver \( R_{x_r} \) is also determined from \( \beta[r] \). At each iteration \( r \), the value \( \beta[r] \) is updated to match the delivery ratio of \( R_{x_r} \) to that of receivers \( R_{x_1}, \cdots, R_{x_{r-1}} \). In the remainder of this section, we will explicitly show any dependencies on \( \beta[r] \), for example \( K^m_r(\beta[r]) \) represents receiver \( R_{x_r} \’s \) mode \( m \) knowledge differential probability given that \( \beta[r] \) is used as the mode vector.

We begin by allocating all transmissions to mode 1, so that \( \beta_1[1] = 1 \) and \( \beta_m[1] = 0 \), for \( m = 2, \cdots, R \). By [24], all mode 1 transmissions are knowledge differential transmissions for \( R_{x_1} \). Therefore the delivery ratio for \( R_{x_1} \) after iteration 1 is \( Q_1[1] = 1 \).
Algorithm 6 Fairness algorithm.

1: Set \( \beta_1[1] = 1 \), and \( \beta_2[1], \ldots, \beta_R[1] = 0 \).
2: for \( r = 2 \) to \( R \) do
3:   Calculate \( Q_{r-1}[r-1] \) and \( Q_r[r-1] \) using (29), substituting \( \beta = \beta[r-1] \).
4:   Calculate \( \omega[r] \) from (37).
5:   Set \( \beta_r[r] = \omega[r] \)
6: for \( m = 1 \) to \( r - 1 \) do
7:   Set \( \beta_m[r] = (1 - \omega[r]) \beta_m[r-1] \)
8: end for
9: end for
10: Set \( \beta = \beta[r] \).

| Setting | \( \mu \) |
|---------|--------|
| A       | [0.8, 0.7, 0.6, 0.5] |
| B       | [0.9, 0.8, \ldots, 0.5] |
| C       | [0.9, 0.85, \ldots, 0.55] |

In each iteration \( 2 \leq r \leq R \) of the for loop, we find a value of \( \beta[r] \) which will achieve \( Q_1[r] = \cdots = Q_r[r] \), equalising the delivery ratio of \( R_x \), with that of receivers \( R_{x_1} \) to \( R_{x_{r-1}} \). This is determined by reducing the time spent in modes \( 1, \ldots, r-1 \) by a factor of \( (1 - \omega[r]) \) to allow a fraction of time \( \omega[r] \) to be spent in mode \( r \).

By Property 1, this has the effect of scaling \( Q_1[r], \ldots, Q_r[r-1] \) by a factor \( 1 - \omega[r] \), so that

\[
Q_m[r] = (1 - \omega[r])Q_m[r - 1]
\]  
(35)

for all \( m < r \).

Combining (27), (29), Property 1, and (24), the delivery ratio of receiver \( R_x \) in iteration \( r \) is given by

\[
Q_r[r] = \frac{\sum_{m=1}^{r} \beta_m[r] R_m \beta[r]}{1 - B_r[r]}
= (1 - \omega[r])Q_r[r - 1] + \frac{\omega[r]}{1 - B_r[r]}.
\]  
(36)

Therefore to equalise the delivery ratios of \( R_x \) and \( R_{x_{r-1}} \) the task is to find the value of \( \omega[r] \) for which \( Q_r[r] = Q_{r-1}[r] \). This can be solved using (35) and (36) to give

\[
\omega[r] = \frac{Q_{r-1}[r-1] - Q_r[r - 1]}{Q_r[r - 1] - Q_{r-1}[r - 1] + \frac{1}{1 - B_r[r]}}.
\]  
(37)

For \( m = 1, \ldots, r - 1 \) the new values of \( \beta_m[r] \) are now given by

\[
\beta_m[r] = \begin{cases} 
(1 - \omega[r])\beta_m[r - 1], & m < r \\
\omega[r], & m = r \\
0, & m > r 
\end{cases}
\]  
(38)

This process is repeated for each receiver to equalise the delivery ratio of each subsequent receiver, until all receivers are equal. After this process is complete, the amount of time that should be spent in each mode \( m \) to achieve fairness is given by

\[
\beta = \beta[R].
\]  
(39)

The delivery ratio of every receiver \( R_x \) using the mode vector \( \beta \) will be equal. Substituting (24) and (26) into (27), the delivery ratio of \( R_{x_1} \) is found to be \( \mu_1 \beta_1 \), so by (29), the delivery ratio of \( R_{x_1} \) is \( Q_1 = \beta_1[R] \). Therefore, using the mode vector \( \beta \) will give the delivery ratios

\[
Q_1 = Q_2 = \cdots = Q_R = \beta_1[R].
\]  
(40)

C. Simulation results

We can now compare the delivery rates achieved both in theory and in practice. The channel rates for each setting we will investigate is given in Table VII.
Fig. 8. Calculated delivery rates achieved after each iteration \( r \) of Algorithm 6 under Setting A from Table VII. These are compared against \( \mu \), the channel rate for each receiver.

In Fig. 9 we compare the calculated and simulated delivery rates under the values of \( \beta \) determined from Algorithm 6. Similar to Fig. 7 we observe that for low receiver indices, estimates match well with calculated values, while some loss of accuracy is observed for higher receiver indices.

IX. CONCLUSION

In this paper we have demonstrated that, by relaxing the complete information acquisition requirement, it is possible to exceed the min-cut capacity of a broadcast packet erasure network. By implementing a deterministic coding scheme, we were
able to achieve multi-rate delivery of the same set of ordered information to a set of heterogeneous receivers, allowing the sender’s transmission rate to exceed the channel rate of a number of receivers.

Despite the complexity associated with the analysis of deterministic coding schemes, through the use of some approximations we were able to obtain tractable estimates of the receivers’ delivery rates. These estimates were shown to be reasonably accurate, although some loss of accuracy was observed with a larger number of receivers. Based on our approximate analysis, we introduced a fairness algorithm that determined how the sender should allocate its resources to achieve fair delivery rate performance for the receivers.

While the rate analysis we have presented in this paper lends insight into the process of multi-rate delivery, it is specific to the network coding scheme we have used. It remains unclear whether higher delivery rates could in fact be achieved using other deterministic transmission schemes. Future work should include the characterisation of the capacity of a broadcast network, given the relaxation of the complete information acquisition requirement.

REFERENCES

[1] R. Ahlswede, N. Cai, S. Li, and R. Yeung, “Network information flow,” IEEE Trans. Inf. Theory, vol. 46, no. 4, pp. 1204–1216, 2002.
[2] A. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, “Capacity of wireless erasure networks,” IEEE Trans. Inf. Theory, vol. 52, no. 3, pp. 789–804, 2006.
[3] D. Lun, M. Médard, and R. Koetter, “Network coding for efficient wireless unicast,” in Proc. International Zurich Seminar on Communications (IZS), 2006, pp. 74–77.
[4] N. Shalman, “Communication over an unknown channel via common broadcasting,” Ph.D. dissertation, Tel Aviv University, 2003.
[5] S. Sanghavi, “Intermediate performance of rateless codes,” in Information Theory Workshop, 2007, ITW ’07, IEEE, 2007, pp. 478–482.
[6] A. Beimel, S. Dolev, and N. Singer, “Rt oblivious erasure correcting,” IEEE/ACM Trans. Netw., vol. 15, no. 6, pp. 1321–1332, 2007.
[7] E. Martinian, “Dynamic information and constraints in source and channel coding,” Ph.D. dissertation, Massachusetts Institute of Technology, 2004.
[8] S. Li, R. Yeung, and N. Cai, “Linear network coding,” IEEE Trans. Inf. Theory, vol. 49, no. 2, pp. 371–381, 2003.
[9] J. Sundararajan, P. Sadeghi, and M. Médard, “A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay,” in Proc. Workshop on Network Coding, Theory, and Applications (NETCOD), 2009, pp. 1–4.
[10] K. Sundararajan, D. Shah, and M. Médard, “ARQ for network coding,” in Proc. IEEE International Symposium on Information Theory (ISIT), 2008, pp. 1651–1655.
[11] B. Swapna, 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.
[12] 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.
[13] D. Luscan, 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.
[14] C. Gkantsidis and P. Rodriguez, “Network coding for large scale content distribution,” in Proc. IEEE Conference on Computer Communications (INFOCOM), vol. 4, 2005.
[15] 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.
[16] 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.
[17] P. Chou, Y. Wu, and K. Jain, “Practical network coding,” in Proc. Annual Allerton Conference on Communication, Control and Computing, Oct. 2003.
[18] 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.
[19] S. Sorour and S. Valaee, “On minimizing broadcast completion delay for instantly decodable network coding,” in Proc. IEEE International Conference on Communications (ICC), May 2010, pp. 1–5.
[20] 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.
[21] 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.
[22] S. Sorour and S. Valaee, “Minimum broadcast decoding delay for generalized instantly decodable network coding,” in Proc. IEEE Global Telecommunications Conference (GLOBECOM), IEEE, 2010, pp. 1–5.
[23] W. Zeng, C. Ng, and M. Médard, “Joint coding and scheduling optimization in wireless systems with varying delay sensitivities,” in IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012, pp. 416–424.
[24] M. Durvy, C. Fragouli, and P. Thiran, “Towards reliable broadcasting using ACKs,” in IEEE International Symposium on Information Theory (ISIT), 2007, pp. 1156–1160.
[25] J. K. Sundararajan, D. Shah, M. Médard, S. Jakubczak, M. Mitzinnenacher, and J. Barros, “Network coding meets TCP: Theory and implementation,” Proc. IEEE, vol. 99, no. 3, pp. 490–512, 2011.
[26] 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.
[27] A. Fu, P. Sadeghi, and M. Médard, “Dynamic rate adaptation for improved throughput and delay in wireless network coded broadcast,” IEEE/ACM Trans. Netw., to appear. [Online]. Available: [http://arxiv.org/abs/1208.3806]
[28] F. Wu, C. Hua, H. Shan, and A. Huang, “Reliable network coding for minimizing decoding delay and feedback overhead in wireless broadcasting,” in IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), 2012, pp. 796–801.
[29] D. S. Lun, M. Médard, R. Koetter, and M. Effros, “On coding for reliable communication over packet networks,” Physical Communications, vol. 1, no. 3, pp. 2–30, 2008.
[30] N. Sundaram, P. Ramanathan, and S. Banerjee, “Multicasting media streaming using network coding,” in Proc. Annual Allerton Conference on Communication, Control, and Computing, September 2005.
[31] T. Cover, “Broadcast channels,” IEEE Trans. Inf. Theory, vol. 18, no. 1, pp. 2–14, 1972.
[32] C. Fragouli, J. Le Boudec, and J. Widmer, “Network coding: An instant primer,” Computer Communication Review, vol. 36, no. 1, p. 63, 2006.
[33] M. A. Artin, Algebra, 1st ed. Prentice-Hall, 1991.