Throughput and Fairness-Aware Dynamic Network Coding in Wireless Communication Networks

Pouya Ostovari and Jie Wu
Department of Computer & Information Sciences, Temple University, Philadelphia, PA 19122

***Abstract*—Network coding techniques have received a lot of attention from the research community for providing reliable broadcasting in error-prone wireless networks. The most common network coding approach is segment coding, in which the packets are partitioned into segments, and linear network coding is performed inside each segment. In order to increase the throughput of network coding and decrease the decoding delay, dynamic coding schemes have been recently proposed. However, these methods incur many feedback messages. In this paper, we propose two dynamic network coding schemes that achieve the maximum throughput and reduce the number of required feedback messages. Moreover, we propose a fair dynamic network coding scheme that performs a trade-off between the throughput and the fairness in terms of decoding delay and the number of decodable packets at different destination nodes. Our simulation results show that our proposed dynamic network coding method provides the same throughput as the ARQ for Network Coding (ANC) method, with up to 90% less feedback messages. Moreover, our fair dynamic network coding can increase decoding delay fairness by about 80%.**

***Index Terms*—Linear network coding, broadcasting, reliability, dynamic coding, wireless networks, error-prone channel, fairness, decoding delay.**

I. Introduction

Random linear network coding [1] has been used in many recent works [2]–[7] to provide reliability and efficient transmission, especially in error-prone wireless networks. In random linear network coding, coded packets are generated by linearly combining the original packets over a finite field. The coded packets have a form of $\sum_{i=1}^{k} \alpha_i \times P_i$, where P and α are the original packets and random coefficients, respectively. The source node generates and transmits random coded packets and the random coefficient vectors. The destination nodes buffer the received coded packets until they receive k linearly independent coded packets. The destination nodes can use any k linearly independent coded packets to decode and retrieve the original packets. Gaussian elimination can be used to decode the packets by solving a system of linear equations. In this scheme, once the destination nodes can decode the coded packets, they need only send one acknowledgement to stop the source node from sending more.

It is well known that using linear network coding results in decoding delay, especially at the nodes with low channel quality. For this reason, it is typical to divide the packets into segments with a fixed block size, and perform random linear coding inside each block, as shown in Figure 1 (a). The source node keeps transmitting the coded inter-segment packet, until all of the destination nodes receive a sufficient number of coded packets. The source node starts transmitting the coded packets of the next segments, once all of the destination nodes have decoded the current segment.

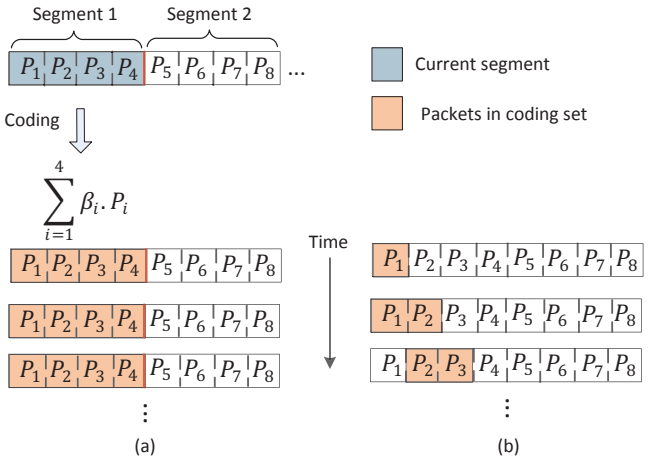


Fig. 1. Segment coding VS. dynamic coding.

Segment coding has two drawbacks when the channel conditions of the receivers are diverse. First, it reduces the throughput, as the sender cannot proceed to the next segment until all of the destination nodes receive the current segment. As a result, the receiver nodes with good channel conditions need to wait for the other nodes, and they will receive useless coded packets. Second, it increases the decoding delay of the nodes with high delivery rates, as the next generation will not be transmitted until all of the nodes retrieve the current generation. That is why dynamic network coding has been studied in [8]–[11]. In dynamic coding, the source node does not limit the coding to a segment, and it performs coding depending on the status of the receiver node.

The source node in the ARQ for Network Coding (ANC) method, which is proposed in [8], combines the first unseen packet of each destination node in each transmission. A node has seen a packet P (original packet) if it can generate a linear combination of the form $P + Q$, using the received coded packets in its buffer. Consider the example in Figure 1 (b). Assume that there are two receiver nodes, d_1 and d_2. In the first time slot, the source node transmits packet P_1, and node d_1 receives it. Then, both of the nodes notify the sender about their status. The sender combines the first packets that have not been seen by each of the nodes, which in this case are

packets P_1 and P_2. Assuming that both of the nodes receive this coded packet, the sender codes P_2 and P_3 in the next transmission.

One of the main drawbacks of dynamic network coding methods is that they incur lots of feedback messages (one feedback from each destination node in ANC after each transmission). Moreover, in the ANC method, the seen packets of the nodes with good channel conditions move faster than the other nodes. Therefore, the nodes with poor channel conditions will experience more decoding delay, which results in decoding and delay unfairness. In this work, we answer the following questions. First, how can we reduce the total number of feedback messages while achieving the maximum possible throughput? Second, how can we use dynamic network coding to provide fairness between the nodes in terms of the number of decodable packets and decoding delay?

The rest of this paper is organized as follows. In Section II, we review the related work. Section III provides the problem definition and setting. We propose our dynamic coding and fair dynamic coding methods in Sections IV and IV-C, respectively. We evaluate the proposed mechanisms in Section V. Section VI concludes the paper.

II. Related Work and Background

The work in [12] introduces Network Coding (NC) for wired networks to solve the bottleneck problem, and [13] shows that NC achieves the capacity for the single multicast session problem. A useful algebraic representation of the linear network coding problem is provided in [14]. The authors in [1] proposed Random linear network coding, and show that randomly selecting the coefficients of the coded packets achieves the capacity asymptotically, with respect to a finite field size.

The works in [15]–[18] address the problem of reliable one-hop broadcasting. In order to provide reliability, the source node needs to retransmit the lost packets by the destination nodes. The source node uses the benefit of network coding in the retransmission phase to improve the transmission efficiency. In order to reduce the number of required retransmissions, these methods combine the packets that have not been received correctly by different receiver nodes.

The authors in [8] address the problem of maximizing throughput in one-hop broadcasting. In order to minimize the buffer size of the sender, the sender needs to drop the packets from its buffer as early as possible. For this purpose, the authors propose the concept of seen packets. The seen packets might not be decodable at the destinations yet, but they are guaranteed to be decodable once the future coded packets are received. In each time slot, the receivers send feedback to notify the sender about their last seen packet. The sender sends a linear combination of the first unseen packets of the destinations. The authors prove that their method achieves the maximum throughput. In order to reduce the decoding delay, a throughput optimal broadcasting method is proposed in [10]. The drawback of this method is that the receivers might not receive the packets in the order of their index.

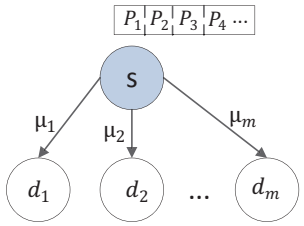


Fig. 2. System setting.

The Systematic online Network Coding (SNC), is proposed in [19]. In this method, a packet that is transmitted by the sender for the first time is sent uncoded. Whenever the receiver node with the most received packets suffers a packet loss, the next packet transmitted by the sender is a linear coded packet, which contains the last unseen packet of each receiver. In order to reduce the worst-case delay for the receiver, the authors propose a second method that considers a threshold for the delays. The sender retransmits a packet in uncoded form, when the remaining time to the deadline of the packet is less than the given threshold.

The adaptive network coding for scheduling real-time traffic with hard deadlines is addressed in [20]. It is assumed that each set of packets that constructs a multimedia frame has a deadline to be received by the destination nodes. As coding all of the packets of a frame together might result in deadline misses, the sender finds the optimal coding size that maximizes the expected gain using dynamic programming and backward induction. After all of the destinations retrieve the current original packets, the sender computes a new coding size based on the remaining time to the deadline of the current frame.

The work in [21] analyzes the delay of dynamic coding schemes. The authors use Markov chain to model the system, in which the statuses of the nodes are the differences between the number of packets at the buffer of the sender and the rank of the receiver's buffer, i.e. zeros state means that the receiver node has already received all of the packets that the sender has in its buffer. The ways in which the next packet can be delivered to a destination node is categorized into zeros state, leader, and chance decoding.

Two moving window network coding methods are proposed in [11], in which the size of the coding window is fixed. The sender tracks the number of received coded packets by each destination node. If the number of received coded packets by all of the receiver nodes is less than the t times a given factor, where t is the current time slot, the sender does not move the coding window. Otherwise, the sender shifts the window. In the second method, the sender moves the coding window with a predefined and fixed velocity. The receiver nodes know the velocity, and will send a NACK message to the sender in the case that they are not able to decode the packet if the sender moves the window. The authors extend their moving window method in [22] to include cooperative transmissions.

III. System Setting

Our model is the same as that in [8]. We consider a single transmitter (e.g., base station), which broadcasts a set of packets $P_1, P_2, ...$ to a set of m receivers (users). The time is divided into slots of equal size, which are synchronized across receivers. The sender can transmit one packet per time slot, and each transmission takes one slot to be delivered to the users. The receivers are connected to the sender via independent erasure channels, and the erasure probability of the link between the sender and the i-th receiver is represented as μ_i. Figure 2 shows the setting.

We focus on random linear network coding, in which random coefficients are used to combine the packets. The receiver nodes store the received coded packets in their buffer, and they are able to decode the coded packet once they receive a sufficient number of coded packets. We assume that immediate and perfect feedback messages are available at the transmitter. The sender uses these feedback messages to decide which packets should be coded and transmitted in the next time slot. In this paper, we refer to the original and coded packets as "packets" and "coded packets," respectively.

The objective in our first proposed method is to maximize the throughput while considering the number of feedback messages. In contrast with [8], where the sender requires a feedback message from each destination node, we want fewer feedback messages to be transmitted at each time slot. In our final proposed method, we also consider fairness, in terms of decoding delay and the number of decodable packets at the destination nodes. Decoding delay of packet P_i is the difference between the generation time of the packet and its decoding time at a destination node. We define the decoding delay unfairness of packet P_j at a given time slot as follows:

$$f'_D(j) = \frac{\sum_{i=1}^{m} |D(i,j) - \bar{D}(j)|}{m} \tag{1}$$

Here, $D(i,j)$ is the decoding delay of packet P_j at receiver node d_i, and $\bar{D}(j)$ represents the average decoding delay of the packet at different receiver nodes. As a result, the decoding delay unfairness of a given packet is the average difference between the average decoding delay of that packet and the decoding delay of the packet at the different receiver nodes. The decoding delay fairness is represented as $f_D(j)$ and defined as:

$$f_D(j) = \frac{1}{f'_D(j)} \tag{2}$$

The decoding delay fairness of a method f_D is the average fairness of the different packets. The decoding unfairness and fairness are as follows:

$$f'_E = \frac{\sum_{i=1}^{m} |E(i) - E|}{m} \tag{3}$$

We represent the total number of decodable packets at the destination nodes and the number of decodable packets at the i-th destination node as E and $E(i)$, respectively. The decoding fairness is the reverse of the decoding unfairness and is represented as $f'_E(j)$:

$$f_E = \frac{1}{f'_E} \tag{4}$$

TABLE I
The set of symbols used in this paper.

Notation	Definition
d_i	The i-th destination node
p_i	The i-th packet
μ_i	The receiving rate of the i-th destination node
$f_D/f'_D)$	The decoding delay fairness/unfairness
$f_D(j)/f'_D(j)$	The decoding delay fairness/unfairness of the j-th packet
$D(i,j)$	The decoding delay of the j-th packet at receiver d_i
$D(j)$	The average decoding delay of the j-th packet at the receiver nodes
f_E/f'_E	The decoding fairness/unfairness
$E(i)$	The number of decodable packets at receiver node d_i
E	The total number of decodable packets
lb/ub	The index of the first unseen packet by a behind/leader node
L	The number of leader nodes
w	The fairness weight
x	The decision variable

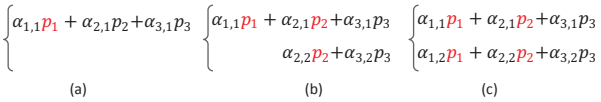


Fig. 3. Examples of *seen* packets.

IV. Dynamic Network Coding

Our dynamic network coding methods are based on the notation of *seen packet*, which is introduced in [8]. A node is said to have seen a packet P (original packet) if it has received a sufficient number of coded packets in its buffer to compute a linear combination of the form $P+Q$. Here, Q is a linear combination of the packets with a greater index than P. Consider Figure 3 (a), in which 3 packets are coded together using random coefficients α_1 to α_3. In this example, P_1 is a seen packet, since, if we consider $\alpha_1 P_1$ as P and $\alpha_2 P_2 + \alpha_3 P_3$ as Q, then the coded packet is a linear combination in the form $P + Q$. However, if we consider P_2 as P, there is no way to compute a linear combination that does not consist of P_1. By the same reasoning, in Figure 3 (b), P_1 and P_2 are seen packets. In Figure 3 (c), if we multiply the first and the second coded packets with $-\alpha_{1,2}$ and $\alpha_{1,1}$, respectively, and add them together we can remove P_1. As a result, in addition to P_1, P_2 is a seen packet. The idea behind a packet seen by a destination node is that the packet is not required to be included in the future coded packets, as it can be decoded by the coded packets received later. Therefore, the sender can drop the packet from its buffer to reduce the buffer size.

In the ANC method, proposed in [8], the sender sends a linear combination of the first unseen packets of each destination node at each time slot. The index of the first unseen packet of a node is 1 plus the index of its last seen packet. Figure 4 (a) shows an example of coding in the ANC method.

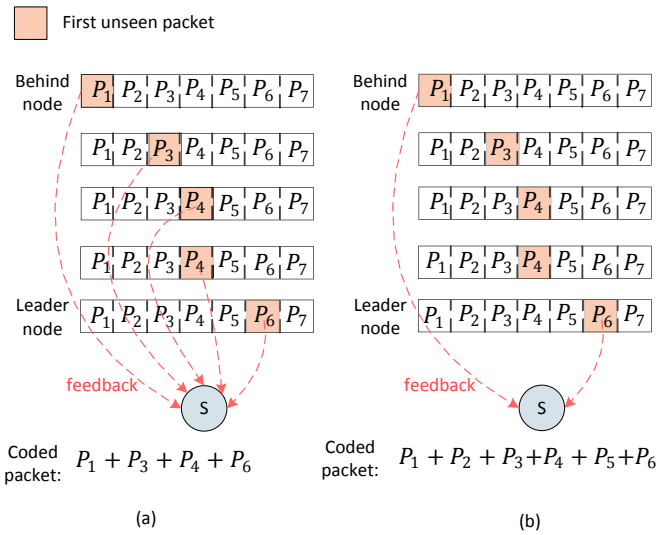


Fig. 4. Coding strategies and required feedbacks. The coefficients are not shown, for simplicity. (a): The ANC method (b): Reduced number of feedbacks.

We do not show the coefficients of the packets for simplicity, and the actual coded packet is $\alpha_1 P_1 + \alpha_3 P_3 + \alpha_4 P_4 + \alpha_6 P_6$. In this way, the sender can drop the packets seen by all of the destinations. In order to inform the sender about the unseen packet, each destination needs to send a feedback message after each transmission by the sender, which incurs many feedback messages.

In contrast to the ANC method, we limit the feedback messages to the leader and behind nodes. We define the *leader* nodes as the nodes whose index of the first unseen packet is the maximum among all of the nodes. In contrast, a *behind* node has the minimum unseen packet index. In Figure 4, the first and the last nodes are behind and leader nodes, respectively. The idea behind our scheme is that we can simply code all of the packets that their index lies in the range of unseen packet by the leader and behind nodes together. In this way, there is no need for receiving feedback messages from the non-leader and non-behind nodes, and we can remove many unnecessary feedback messages.

Our dynamic network coding method works as follows. The source transmits the first packet, and the leader and behind nodes specify their first unseen packet using a feedback message. Then, the source node combines the unseen packets of the destination nodes and transmits a coded packet in the form of $\sum_{i=lb}^{ub} \alpha_i \times P_i$, where lb and ub are the indices of the first unseen packets with the smallest and largest indices, respectively. Symbol α_i represents a random coefficient.

When the source node receives feedback from each destination node, it will have exact information about the first unseen packets by the destination node. Consider Figure 5 (a). The first unseen packets by each destination node are shown with colored cells. In the ANC scheme, after each transmission, 5 feedback messages are required by the source node. After receiving the feedback messages, the source node codes the

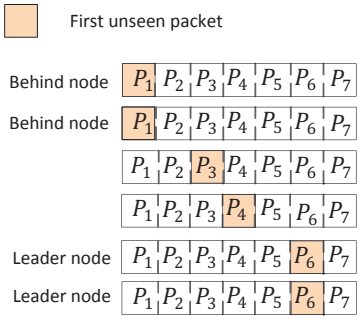


Fig. 5. Multiple behind and leader nodes.

first unseen packets of each destination node together. Now consider Figure 5 (b). If we code all of the packets in the range of the first unseen packet by the behind and the leader nodes, the number of feedback messages will be reduced to 2 messages. Consider Figure 4 (b). In contrast with the ANC approach, only the leader and behind nodes transmit their first unseen packets, and the source node codes all of the packets with an index in the range of these two packets. It should be noted that, as the number of destination nodes increase, the number of non-leader and non-behind nodes increases, which increases the efficiency of our approach.

In the case of multiple leaders or behind nodes, sending the feedback messages might be a challenge. In the following sections, we propose two feedback mechanisms based on whether or not overhearing is possible among the receivers.

A. Dynamic Network Coding without Overhearing

In the case that overhearing is not possible between the receiver nodes, all of the leader receiver nodes need to transmit a feedback message. The nodes can easily check if they are a leader node. When a node receives a packet, it checks the index of its first unseen packet with the indices of the packets included in the received coded packet. If the index of its first unseen packet is equal to the index of the packet with the largest index that is included in the received coded packet, then the node is a leader node. A receiver node that has not received the last transmission cannot be a leader node. The reason is that, when a leader node sends a feedback, the sender will move the coding window to the right by adding a new packet. Therefore, in the following time slots, none of the nodes that miss the new transmission can become a leader node.

In order to find the index of the first packet that should be included in the coding set, the behind nodes that do not receive the current transmission should send a feedback to the sender. It is clear that if a behind node does not receive the current transmission, it will still be a behind node, and should send a feedback message. In the case that all of the behind nodes receive the current transmission, all of them are still behind nodes. However, they do not know it, so they do not send any feedback messages. In this case, the sender will set $lb(t) = lb(t-1) + 1$, where $lb(t-1)$ is the lb in the previous time slot $t-1$. Algorithms 1 and 2 show the receiver and

Algorithm 1 DNC (For receiver d_i)

Initialize: $u(i) = 1$
if d_i received the current transmission **then**
  Set ub to the largest index included in the received coded packet
  if $u(i) = ub$ **then**
    $u(i) = u(i) + 1$
    Node d_i is a leader node. Transmit $u(i)$ to the sender.
  else
    $u(i) = u(i)+1$ //the node might be a middle or behind node
if d_i did not receive the current transmission **then**
  if d_i was a behind node at slot $t-1$ **then**
    Node d_i is a behind node. Transmit $u(i)$ to the sender

Algorithm 2 DNC (Sender side)

Initialize: $lb(1) = 1$, $ub(1) = 1$.
Set $ub(t)$ to the index of unseen packets by the leader nodes.
if Did not receive feedback from a behind node **then**
  $lb(t) = lb(t-1) + 1$
else
  $lb(t) = lb(t-1)$
transmit $\sum_{j=lb(t)}^{ub(t)} \alpha_j b_j$

sender side processes, respectively. The first unseen packet of the i-th node is represented as $u(i)$.

B. Dynamic Network Coding with Overhearing

When the nodes can overhear each other, we can reduce the total number of feedback messages to two per time slot: one from a node in the set of leaders, and one from a behind node. For each leader node, we set a back-off time based on the erasure rate of the nodes. During the back off time, the receiver nodes should listen to the channel. When a leader node finishes its waiting time, it sends its feedback message if it has not overheard feedback from the other leader nodes. As a result, the leader node with the smallest back-off time sends its feedback when its back-off time expires. The same mechanism is used for the set of behind nodes.

The behind nodes that have received the last transmissions do not need to transmit a feedback; however, they might still be a behind node in the case that all of the behind nodes receive the current transmission. Therefore, only one of the nodes that was a behind node in the previous time slot, and missed the last transmission in the current time slot, should send a feedback message to force the sender to keep transmitting the packet with the smallest index in the coding set. The details of the receiver process are provided in Algorithm 3. The sender side algorithm is similar to that of the DNC method.

C. Fair Dynamic Network Coding Scheme

It is proven in [8] that the ANC method achieves the maximum throughput, as each transmission has innovative information for all of the nodes. In a similar way, it can be

Algorithm 3 DNC-OH (For receiver d_i)

Initialize: $u(i) = 1$
if d_i received the current transmission **then**
  Set ub to the largest index included in the received coded packet
  if $u(i) = ub$ **then**
    $u(i) = u(i) + 1$
    Node d_i is a leader node. Wait for a random time
    if did not overhear a feedback from a leader node **then**
      Transmit $u(i)$ to the sender.
  else
    $u(i) = u(i)+1$ //the node might be a middle or behind node
if d_i did not receive the current transmission **then**
  if d_i was a behind node at the previous slot **then**
    Node d_i is a behind node.
    Wait for a random time
    if did not overhear a feedback from a behind node **then**
      Transmit $u(i)$ to the sender.

proven that our proposed schemes in Section IV achieve the maximum throughput. However, all of these three methods have a drawback. The nodes with low error rates receive more coded packets than the other nodes, and become the leaders. The leaders, which have the unseen packet with the largest index among the other nodes, can decode all of the packets that are included in the coded packets. Thus, the decoding delay at these nodes is smaller than the non-leader nodes. The leader nodes force the sender to add new packets in each time slot. As a result, the nodes with higher error rates might not be able to decode the packet for a long time, and they will experience high decoding delays, which result in an unfair delay experience at different receiver nodes.

In order to solve this problem, we propose the Fair Dynamic Network Coding (FDNC) scheme. Most like the DNC method, the sender receives feedback messages from the leader and behind nodes after transmitting a coded packet. However, unlike the DNC method, which adds a new packet to the coding window once a leader receives the last transmission, the sender in the FDNC scheme uses the following equation to decide whether to add a new packet or not:

$$x = (1 - w) \times L - w \times (m - L) \tag{5}$$

Here, m and L are the number of nodes and leader nodes, respectively. The notation w is the fairness weight, and shows the importance of fairness against the throughput. In the case that x is greater than zero, the sender adds the next original packet to the coding set; otherwise, it postpones adding the next packet to the following transmission slots. If we set w to 0, the sender will add a new packet to the coding set once a leader node receives the current transmission; therefore, the FDNC will work similarly to the DNC method. On the other hand, for $w = 1$, the sender will not add a new packet until

Algorithm 4 FDNC Algorithm (Sender side)

Initialize: $lb(1) = 1$, $ub(1) = 1$.
Receive feedback from leader and behind nodes
Set $ub(t)$ to the index of unseen packets by the leaders
if Did not receive feedback from a behind node **then**
  $lb(t) = lb(t-1) + 1$
else
  $lb(t) = lb(t-1)$
$x = w \times L - (1-w) \times m - L$
if $x \geq 0$ **then**
  $ub(t) = ub(t-1) + 1$
transmit $\sum_{j=lb(t)}^{ub(t)} \alpha_j b_j$

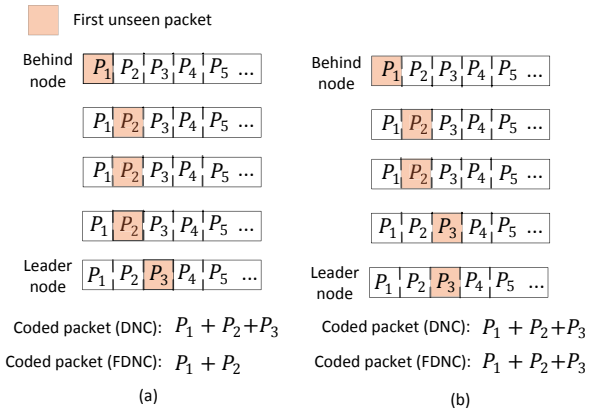


Fig. 6. The FDNC method VS. DNC. The coefficients of the coded packets are not shown, for simplicity.

all of the nodes become a leader node. The sender's protocol is shown in Algorithm 4. The receivers' algorithm is similar to that of the DNC method.

Consider Figure 6 (a), and assume that w is equal to 0.4. In this case, we have a leader node and 4 non-leader nodes; thus, we have $x = (1 - 0.4) - 0.4 \times 4 = -1$. As a result, when we use the FDAN method, the sender does not add packet P_3 into the coding set, although the first unseen packet by the leader node is P_3. The coded packet in the next transmission will be $\alpha_1 P_1 + \alpha_2 P_2$. On the other hand, the DNC and ANC methods add codes P_1, P_2, and P_3 together. The FDNC method achieves more fairness by delaying adding P_3 until more destination nodes reach the leading state. Now consider Figure 6 (b). In this case, we have 2 leaders and $x = (1 - 0.4) \times 2 - 0.4 \times 3 = 0$. Consequently, both the ANC and FDNC methods transmit $\alpha_1 P_1 + \alpha_2 P_2 + \alpha_3 P_3$. In Figure 6, the coefficients of the coded packets are not shown, for simplicity.

V. Simulations

In this section, we evaluate our proposed mechanisms DNC (Dynamic Network Coding), DNC-OH (Dynamic Network Coding with Overhearing), and FDNC (Fair Dynamic Network Coding). We compare our methods with the proposed schemes in [8] and [11]. For this purpose, we implemented a simulator in the MATLAB environment. We run the simulations on 100 random topologies, with different link error rates. The plots in this paper are based on the average outputs of the simulation runs. In the simulations, we assume the existence of reliable feedback messages. Also, the receiver nodes are synchronized with the sender. The metrics that are evaluated in our simulations are as follows:

- Throughput: we define the throughput as the average number of innovative packets received by the destination nodes in each time slot.
- Number of decodable packets: we measure the total number of decodable packets by the destination nodes after 10 time slots.
- Decoding delay: the decoding delay of an original packet at a given node is the time that it takes the node to decode the packet.
- Decoding fairness f_E (see Equations (4) and (3)).
- Delay fairness f_D (see Equations (2) and (1)).

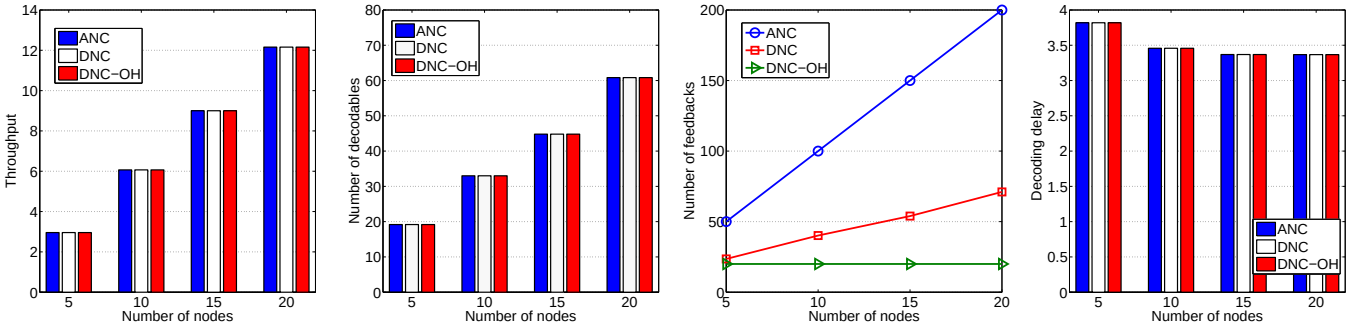
A. Simulation Results

We first compare the DNC and DNC-OH methods with the proposed method in [8], which we refer to as the ANC method. Then, we evaluate the performance of the FDNC against the ANC and MW [9] schemes.

1) Dynamic Network Coding: the purpose of the simulations in this section are to show that the DNC and DNC-OH methods achieve the same throughput, number of decodable packets, and decoding delay as the ANC method, with a fewer number of feedback messages.

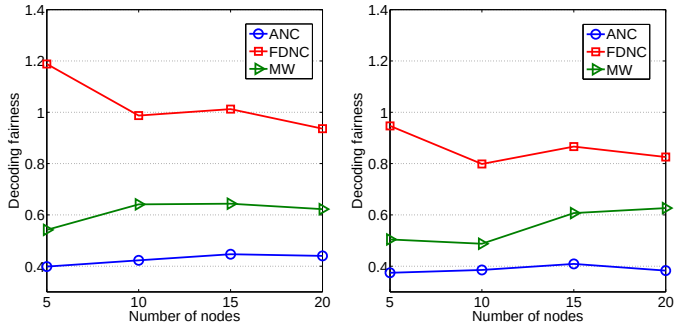
In the first experiment, we evaluate the throughput. As depicted in Figure 7 (a), the throughput of the DNC, DNC-OH, and ANC methods are the same. Similarly, Figure 7 (b) shows that the number of decodable packets in the DNC, DNC-OH, and ANC methods are the same. The reason is that, in all of the methods, the first unseen packet of the destination nodes are added to the transmitted coded packet. As expected, the throughput and the number of decodable packets increases in Figures 7 (a) and (b) as we increase the number of nodes, which is due to more receivers.

We next evaluate the number of feedback messages. Figure 7 (c) shows the total number of transmitted feedbacks. As expected, the ANC method has the largest number of feedback messages, since in each time slot, all of the nodes need to inform the source node about their first unseen packet. The DNC-OH method has the fewest number of feedback messages, since the source node receives just two feedbacks from a leader and a behind node. That is why the number of feedback messages does not increase as we increase the number of nodes. In the DNC method, the nodes cannot overhear each other; as a result, all of the leader and behind nodes transmit a feedback messages. Consequently, the number of feedback messages in the DNC method is more than that of the DNC-OH method. This figure shows that the number of feedback messages in the DNC-OH and DNC methods are up to 90% and 65% less than the ANC method, respectively.



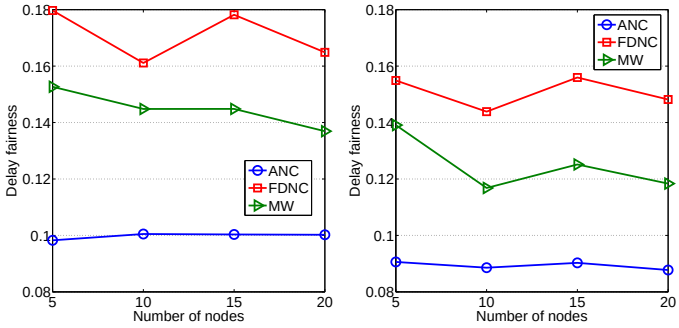
(a) (b) (c) (d)

Fig. 7. Comparison between the DNC, DNC-OH, and ANC methods, $\mu \in [0.3, 0.9]$. (a): Throughput, (b): Number of decodable packets (c): Number of feedback messages, (d): Decoding delay.



(a) (b)

Fig. 8. Decoding fairness, $w = 0.7$. (a): $\mu \in [0.3, 0.9]$ (b): $\mu \in [0.4, 1]$.



(a) (b)

Fig. 9. Delay fairness, $w = 0.7$. (a): $\mu \in [0.3, 0.9]$ (b): $\mu \in [0.4, 1]$.

Figure 7 (d) shows the average decoding delay of the packets. It can be inferred from the figure that the decoding delay of the packets in the DNC, DNC-OH, and ANC methods are equal. It should be noted that, in computing the decoding delay, we do not consider the packets that are not decodable. As a result, when we increase the number of receiver nodes, the probability of having more nodes with good channel conditions increases. That is why the decoding delay in Figure 7 (d) increases as we increase the number of receiver nodes.

2) Fair Dynamic Network Coding: in this section, we compare the fairness, throughput, delay and number of decodable packets of our proposed FDNC method with the ANC and MW methods. Figure 8 (a) shows the decoding (number of decodable packets) fairness of the FDNC method, ANC and MW methods. It can be inferred from the figure that the decoding fairness of the ANC and MW methods are about 66% and 40% less that that of the proposed FDNC method. The reason is that our method performs a trade-off between fairness and throughput. Figure 8 (b) shows the fairness of the methods when the delivery rate of the links are in the range of $[0.4, 1]$.

We show the delay fairness of the FDNC, ANC, and MW methods in Figure 9 (a). The delay fairness of the FDNC algorithm is about 80% and 20% more than that of the ANC and MW methods. It should be noted that the distortion of the plots is due to the randomness of the topologies. The ANC method does not have much distortion, since the leader nodes can always decode the packets with small decoding delay, and the other nodes experience large decoding delays. We repeat this simulation with more reliable links in Figure 9 (b). We see almost the same pattern in Figures 9 (a) and (b).

In the next experiment, we compare the throughput of the methods and show the results in Figure 10. As mentioned before, the ANC algorithm adds a new packet to the coded packet once a leader receives the last transmission. As a result, each transmitted packet is innovative to all of the receivers. That is why the ANC scheme has the highest throughput in Figure 10. The figure shows that the throughput of our scheme is between that of the ANC and MW methods. We increase the delivery rate of the receiver nodes to the range of $[0.4, 1]$, and repeat the previous experiment. The throughput of the methods in Figure 10 (b) are more than that of in Figure 10 (a), which is due to the existence of more reliable links.

In the last experiment, we compare the number of decodable packets of the FDNC method, ANC and MW. Figure 11 (a) shows that the number of decodable packets in the proposed FDNC method is up to 55% and 30% more that that of the ANC and MW methods. It might be confusing that the number of decodable packets in the FDNC method is more than that of the ANC method, while its throughout is less. The reason is that throughput just tells us the average number of

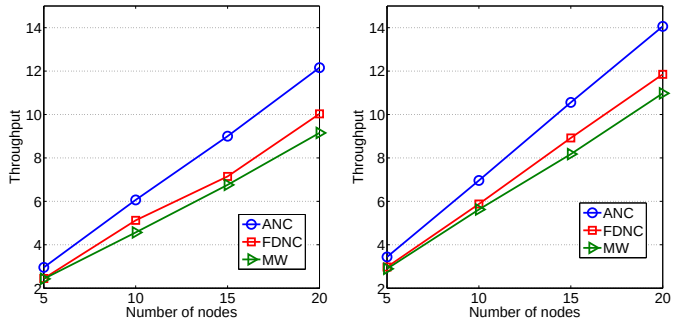


Fig. 10. Throughput, $w = 0.7$. (a): $\mu \in [0.3, 0.9]$ (b): $\mu \in [0.4, 1]$.

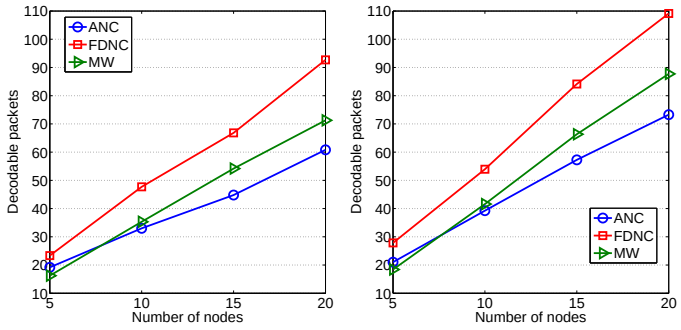


Fig. 11. Number of decodable packets, $w = 0.7$. (a): $\mu \in [0.3, 0.9]$ (b): $\mu \in [0.4, 1]$.

innovative received packets per time slot. A non-leader node might have some linearly independent packets in its buffer, but they might not be decodable at the time we measure the number of decodable packets of the destination nodes. Figure 11 (b) shows the study with more reliable links in the range of $[0.4, 1]$, which results in a larger number of decodable packets at the destination nodes.

VI. Conclusion

In network coding, the packets to be sent are usually divided into segments, and network coding is performed inside each segment. However, in this technique, throughput is dominated by the receivers with the worst channel conditions. In order to increase the throughput of network coding, dynamic coding schemes have recently been proposed, which incur many feedback messages. In this work, we propose two dynamic network coding methods, the DNC and DNC-OH methods, to reduce the number of feedback messages. Moreover, in order to provide decoding delay and number of decodable packets fairness, we propose the FDNC method, which provides a trade-off between the throughput and the fairness. Our simulation results show that the DNC and DNC-OH methods provide the same throughput as the ANC method, which is a leading dynamic coding method, with about 90% less feedback messages. Moreover, the FDNV method can increase decoding fairness by about 80%, while sacrificing 15% of throughput.

Acknowledgment

This research was supported in part by the NSF grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.

References

[1] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[2] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: practical wireless network coding," in *ACM SIGCOMM*, 2006.

[3] D. Koutsonikolas, C. Wang, and Y. Hu, "CCACK: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1–9.

[4] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *ACM SIGCOMM*, 2007.

[5] P. Ostovari, J. Wu, and A. Khreishah, *Network Coding Techniques for Wireless and Sensor Networks*. Springer, 2013.

[6] P. Ostovari, A. Khreishahand, and J. Wu, "Efficient symbol-level transmission in error-prone wireless networks with network coding," in *IEEE WoWMoM*, Jun 2013.

[7] A. Khreishah, I. Khalil, P. Ostovari, and J. Wu, "Flow-based xor network coding for lossy wireless networks," *IEEE Transactions on Wireless Communications*, vol. 11, no. 6, pp. 2321–2329, 2012.

[8] J. K. Sundararajan, D. Shah, and M. Médard, "ARQ for network coding," in *IEEE ISIT*, 2008, pp. 1651–1655.

[9] A. Fu, P. Sadeghi, and M. Medard, "Dynamic rate adaptation for improved throughput and delay in wireless network coded broadcast," *arXiv*, 2012.

[10] J. K. Sundararajan, P. Sadeghi, and M. Médard, "A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay," in *IEEE NetCod'09*, 2009, pp. 1–6.

[11] F. Wu, C. Hua, H. Shan, and A. Huang, "Reliable network coding for minimizing decoding delay and feedback overhead in wireless broadcasting," in *IEEE PIMRC*, 2012, pp. 796–801.

[12] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[13] S. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[14] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782– 795, Oct 2003.

[15] L. Lu, M. Xiao, M. Skoglund, L. Rasmussen, G. Wu, and S. Li, "Efficient network coding for wireless broadcasting," in *Wireless Communications and Networking Conference (WCNC)*, 2010, pp. 1–6.

[16] L. Lu, M. Xiao, and L. Rasmussen, "Relay-aided broadcasting with instantaneously decodable binary network codes," in *Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN)*, 2011, pp. 1–5.

[17] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 2, pp. 914–925, 2009.

[18] W. Fang, F. Liu, Z. Liu, L. Shu, and S. Nishio, "Reliable broadcast transmission in wireless networks based on network coding," in *Computer Communications Workshops (INFOCOM WKSHPS)*, 2011, pp. 555–559.

[19] J. Barros, R. A. Costa, D. Munaretto, and J. Widmer, "Effective delay control in online network coding," in *IEEE INFOCOM*, 2009, pp. 208–216.

[20] L. Yang, Y. E. Sagduyu, and J. H. Li, "Adaptive network coding for scheduling real-time traffic with hard deadlines," in *ACM MobiHoc*, 2012, pp. 105–114.

[21] A. Fu, P. Sadeghi, and M. Médard, "Delivery delay analysis of network coded wireless broadcast schemes," in *IEEE WCNC*, 2012, pp. 2236–2241.

[22] F. Wu, C. Hua, H. Shan, and A. Huang, "MWNCast: cooperative multicast based on moving window network coding," in *IEEE Globecom*, 2012.