Cooperative multicast is an effective solution to address the bottleneck problem of single-hop broadcast in wireless networks. By incorporating with the random linear network coding technique, the existing schemes can reduce the retransmission overhead significantly. However, the receivers may incur large decoding delay and complexity due to the batch decoding scheme. In addition, the dependency on the explicit feedback leads to scalability problem in larger networks. In this paper, a cooperative multicast protocol named MWNCast is proposed based on a novel moving window network coding technique. We develop analytical models and show three properties of the proposed scheme. Firstly, without explicit feedback, the packet recovery loss probability of the receivers drops almost exponentially with the increase of window size. Secondly, the average decoding delay of a receiver is upper bounded by $O\left(\frac{1}{q}\right)$ asymptotically with respect to its traffic intensity $q$. Thirdly, given the target throughput, the decoding complexity of MWNCast scales as $O(W)$ as the window size $W$ increases. Simulation results show that MWNCast outperforms the existing schemes by achieving better tradeoff between the throughput and decoding delay, meanwhile keeping the packet recovery loss probability and decoding complexity at a very low level without explicit feedback.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Due to the broadcast nature of wireless channels, wireless networks have been deemed as an efficient solution for multicast file delivery, multimedia streaming services, etc. Under perfect channel conditions, multiple clients within the transmission range of a single transmitter node can receive the same piece of data simultaneously without incurring any extra overhead. However, this assumption is invalid in practice since wireless channels are subject to fading due to signal attenuation, shadowing and multipath effects, leading to random failure of packet reception at different clients.

Although packet error can be tolerated to some extents in most multimedia streaming applications, excessive packet losses are unacceptable because it can lead to the degradation of quality of experience (QoE) to the end users. In order to improve the reliability of multicast, many techniques and protocols have been developed. One class of solutions follow the error recovery path that try to tackle the packet loss problem using the automatic repeat
request (ARQ) or combined with forward error correction (FEC) (e.g., [2,3]), which however lead to feedback storm problem since the source node relies on the feedback from clients to make retransmission decisions. To address this issue, another class of schemes adopt the rateless coding strategy (e.g., [4–6]), whereby the source node keeps transmitting coded symbols without explicit feedback, and any clients can decode the packet after accumulating enough symbols. Although such approaches are able to provide reliable transmissions, they may suffer from the bottleneck problem, that is, the throughput of the overall system is limited by the node with the worst channel capacity.

As a natural solution to the bottleneck problem in multicast, cooperative communications have drawn increasing attentions recently. In [7], integrated with layered video coding and packet level forward error correction, the randomized distributed space time codes are adopted to design cooperative multicast scheme that can provide efficient and robust video delivery. Relay selection has been studied in [8] to improve the performance of cooperative multicast in a mobile computing environment. The outage probability with cooperative multicast is analyzed in [9], which suggests that the performance can be improved with more relay nodes. These schemes demonstrate the effectiveness of physical-layer cooperation in alleviating the bottleneck problem in multicast, but they may incur some difficulties in practical implementation, such as tight time synchronization. Furthermore, the sequential retransmissions of the lost packets to multiple receivers (requested by feedback) can reduce the bandwidth efficiency.

One potential way to address this issue is to utilize network coding techniques whereby the lost packets can be encoded together to reduce the number of retransmissions. For example, [10] shows the benefit of cooperation at the network layer via a simple XOR network coding technique. In [11], the random linear network coding (RLNC) [6] is adopted for multicast applications, and the channel and power allocation in relaying nodes are optimized for maximizing the multicast rate. It is shown in [12] that compared to the physical-layer cooperation, the use of RLNC at the relays can enhance the system throughput. In [13], a RLNC-based opportunistic multicast protocol is proposed which can alleviate the bottleneck problem effectively. However, to avoid throughput degradation, the block size in RLNC has to scale with the number of receivers [14], which in turn leads to large decoding delay and complexity. In addition, the centralized scheduling policies in [10–12] rely on the feedback from the relays and receivers about the packet reception status, which make them difficult to scale to larger network size in practice.

In this paper, a cooperative multicast protocol named MWNCast is proposed based on the moving window network coding (MWNC) technique. By exploiting the residual capacity of relay nodes to serve the bandwidth starving receivers, the proposed scheme can effectively alleviate the bottleneck problem in wireless multicast. Based on the random walk and point process theory, we show three properties of MWNCast. Firstly, without explicit feedback, the packet recovery loss probability of the receivers drops almost exponentially with the increase of window size. Secondly, the average decoding delay experienced by a receiver is upper bounded by the order of $O\left(\frac{1}{(1-\rho^q)}\right)$, where $\rho$ is the traffic intensity of the node. The decoding delay of receivers with diverse channel conditions are fairly balanced. Thirdly, the average decoding complexity scales as $O(W)$ ($W$ is coding window size) for a given target throughput. Together with the first property, this suggests that the increase of the computational complexity for enhancing the reliability is moderate. We provide simulation results to validate the theoretical results, which show that the proposed scheme can not only guarantee the reliable transmission without explicit feedback, but also achieve high throughput with low decoding delay and complexity.

The rest of this paper is organized as follows. In Section 2, the system models assumed in this paper is introduced. We present MWNC in Section 3. In Section 4, an overview of MWNCast is firstly provided, followed by its functional modules in details. In Section 5, we establish the theoretical framework and then prove three key properties of MWNCast. Simulation results are provided in Section 6 and finally we conclude this paper in Section 7.

2. System model

We consider a wireless network consisting of a source node $s$ (or base station interchangeably) and a set of $N$ receivers. The source node has a stream of packets to be transmitted to all receivers. As discussed in previous section, due to the lossy wireless channel, the capacity of plain broadcast (even with a sophisticated network coding scheme) is limited by the worst receiver. To address this problem, we adopt a cooperative networking structure, whereby a subset $R \subseteq \mathcal{N}$ of nodes are selected as relays, which perform not only the normal receiving function to receive data from the source, but also the relaying function that forwards the received data to the remaining subset $\mathcal{E}$ of end receivers ($\mathcal{E} = \mathcal{N} \setminus R$). To simplify the design of protocol, we assume that relay nodes only receive data from the source, while the end receivers can receive data from both the source and the relay nodes.

Similar to [11], we assume that there are $K$ orthogonal channels that can be operated by each node. Therefore, in order to avoid co-channel transmission interference between the source and the relay nodes, at most $K - 1$ relay nodes are allowed to transmit concurrently with the source. Time is divided into slots, and each node is equipped with one half-duplex radio, so a relay node cannot receive and relay at the same time.

The wireless channel between the nodes are modeled as memoryless erasure channel. For a packet sent by node $i \in R \cup \{s\}$, node $j \in \mathcal{N}$ has probability $c_{ij}$ to receive it successfully, where $c_{ij}$ denotes the packet reception probability (PRP) of the link between nodes $i$ and $j$. Similar to [15], successful packet receptions in different time slots and receivers are considered to be independent. In this paper, we assume the PRPs of all links in the network are stationary and collected by the source node through some online or offline measurements [16, 17]. Note that the overhead of

---

1 We assume frequency division multiple access (FDMA) in this paper, but it can be easily generalized to time division multiple access (TDMA) too.
collecting channel statistics, i.e., PRPs, could be much lower than that of collecting explicit feedback from receivers if the dynamics of the network are relatively low.

Note that \( c_{ij} \) is equivalent to the capacity of link \((i,j)\) since it is the maximum achievable throughput for error-free transmission from node \(i\) to \(j\). In the following, without abusing the notation, we refer to \( c_{ij} \) the PRP as well as the capacity of the link. In particular, let \( c_{ij} \) denote the link capacity from the source to any node \(j \in \mathcal{N} \).

3. Moving window network coding

To improve the performance of wireless multicast, many different network coding techniques have been proposed from different perspectives. The random linear network coding scheme [6] adopts a block transmission strategy which can approach the capacity with less feedback overhead. Unfortunately, it is shown in [14] that the block size of RLNC has to scale with the increase of the number of receivers to avoid the loss of throughput, which however will result in large decoding delay. The ARQ-based online network coding (ANC) [18] achieves the one-hop maximum multicast throughput, but the decoding delay of the receivers with worse channel conditions is unfairly large. Many solutions have been proposed for this problem (e.g., [19–24]). When the number of receivers is small, the schemes proposed in [19,20] can reduce the decoding delay, but the optimal throughput and decoding delay cannot be achieved simultaneously for larger network size. In [21], a delay threshold based on scheme is proposed to incorporate with the ANC scheme, which can guarantee the decoding delay to be within the prescribed bound at the cost of throughput degradation. The instant decodable network coding can effectively minimize the decoding delay, but it cannot guarantee the order of decoding [22]. In [23,24], the instant decodable network coding is generalized so that hard delay constraint for each packet could be satisfied, but their schemes are optimal only when there are no more than three receivers. Note that most of these delay control schemes rely on the feedback from receivers. With virtually no feedback information, the optimal RLNC strategy for delay-constrained traffic is studied in [25], but the scheme still suffers from the throughput degradation problem of RLNC when the network scale increases.

Motivated by these techniques, we propose the MWNC scheme [26] to combine the advantageous features of traditional network coding schemes. In the following, we use \( p_i \) to denote the \(i\)th original packet, and \( s_t \) to denote the network coding symbol to be broadcasted at the base station in slot \(t\). For clarity, we only discuss MWNC at the base station here and will further generalize its definition for the relay stations in Section 4. The MWNC adopts the encoding strategy similar to RLNC, but the block of packets to be encoded in each slot is moving forward at a constant speed \(V\) packet/slot (see Fig. 1). Specifically, in time slot \(t\), a block of \(W\) packets with the sequence number ranging from \([V \cdot t] - W + 1\) to \([V \cdot t]\) are encoded with random coefficients in a finite field \(G(2^3)\) at the base station.\(^2\)

\[
s_t = \sum_{i=\max\{1, [V \cdot t] - W + 1\}}^{\min\{V \cdot t\}} x_{ti} \times p_i,
\]

where the coefficient \(x_{ti}\) is uniformly distributed in \(\{G(2^3)\setminus \{0\}\}\). The coefficients \(x_{ti}, i = [V \cdot t] - W + 1, \ldots, [V \cdot t]\), which are sent along with coded symbol \(s_t\), are often considered as the overhead of network coding technique [6,18].

From the successfully-received coded symbols, the receiver attempts to decode the original packets through Gaussian elimination approach (see [27]). In Gaussian elimination, the coded symbols are treated as linear equations, and the associated coefficient matrix is transformed into an upper triangular matrix (row echelon form) by elementary row operations. A typical example of the decoding process is shown in Fig. 2, where Gaussian elimination is performed progressively as the coded symbols are received. Finally the original packets can be retrieved when the reduced matrix has full rank (Fig. 2(b)\(^3\)). Note that \(V\) packet/slot represents the target throughput, so it should be within the network capacity.

To facilitate the understanding of what may happen at a receiver, we restate the definition of a receiver “seeing” a packet, which was originally introduced in [20].

**Definition 1** (A packet is seen). A packet is regarded as “seen” by the receiver if the receiver could express the packet as a linear combination of packets with greater indexes.

In Table 1, we show an example where the window size \(W = 3\) and the moving speed \(V = 0.5\) packet/slot. In this example, the source sends the uncoded packets \(p_1\) twice in the first two time slots, both of which are lost by the receiver. Then it sends coded symbols \(p_1 \oplus p_2\) (\(\oplus\) denotes the linear combination over packet with coefficients which are chosen independently in the next two slots), both of which gets received. From the first combination sent in slot 3, packet \(p_1\) gets “seen” by the receiver, because it can be

\(^2\) \([\cdot]\) is the ceil function to guarantee that the boundaries of the window are aligned to integer values. Note that if \([V \cdot t] < W\), then the block is started from 1 to \([V \cdot t]\).

\(^3\) The first column are removed because the corresponding packet \(p_1\) has been decoded.
expressed by a combination involving $p_2$, although neither $p_1$ or $p_2$ can be decoded. From the second combination in slot 4, the subsequent packet $p_3$ gets “seen”. Since all the packets which participate encoding have been “seen”, both $p_1$, $p_2$ are decoded. From the fifth time slot, a full window of three packets are encoded in each time slot, which is moved forward with the speed of $V = 0.5$ packet/slot. Note that because there is no feedback mechanism in MWNC, it cannot guarantee 100% reliability. For example, $p_3$ and $p_4$ will get lost at the end of the 12th time slot, since the receiver can never gather enough information to decode them after the window has moved to $p_5$, $p_6$, $p_7$, although the receiver has “seen” $p_1$ from the symbol received in the 8th time slot. Besides, the coded symbol received in slot 8th becomes useless and will be discarded by the receiver. However, we will show in Section 5 that the recovery loss probability with MWNC drops almost exponentially with the increase of window size.

MWNC has some other interesting properties. First, the decoding occurs progressively, therefore it avoids the intrinsic decoding delay problem incurred by RLNC. In addition, the decoding opportunity is balanced between clients with good and poor channel conditions, so none of the clients will be dominated by other clients with better channel conditions. Second, the decoding complexity of MWNC is much lower than RLNC due to the moving window strategy. Intuitively, this is because the coding coefficient matrix in buffer becomes sparse after Gaussian elimination (see Fig. 2), so that the row reduction for a newly received symbol does not take many operations. In Section 5, we will develop theoretical models to justify these properties.

Note that the concept of network coding over a moving window has been considered in [28,29]. In [28], RLNC is incorporated with the congestion window in TCP protocol to improve the throughput in the lossy wireless environment. In [29], SlideOR is proposed to encode packets in overlapping window, which can avoid the throughput loss in opportunistic routing. Our scheme differs from these schemes in the following aspects. Firstly, MWNC may achieve better control of the decoding delay and complexity in cooperative multicast by appropriately setting the moving speed and window size, which is made possible by our analytical results. Existing schemes, which were designed for different scenarios, are not directly applicable in cooperative multicast. Secondly, these schemes rely on the feedback of the receivers to move forward the coding window, which is nontrivial in large-scale wireless broadcasting since the ACKs of different receivers have to be carefully scheduled to avoid collision. Even if the reliability of ACKs can be guaranteed, the feedback delay may lead to the degradation of the network throughput. In our scheme, the coding window of size $W$ is moving forward according to a prescribed moving speed $V$ packet/slot. Although there is no explicit feedback from any of the receivers, we show that if $V$ packet/slot is properly set, it is possible to achieve high reliability by properly setting the window size $W$.

4. Design of MWNCast

In this section, we propose MWNCast, a cooperative multicast protocol based on the MWNC technique. Before elaborating on the details of the protocol, we briefly
introduce the motivation and the basic functionalities of MWNCast with a simple example.

4.1. Overview of MWNCast

Consider a simple example as shown in Fig. 3(a), which consists of three receivers. In the figure, the number on each link is the corresponding PRP. For plain broadcast, it is easy to see that the capacity (packet/slot) of the system is 0.4 packet/slot due to the bottleneck receiver $R_3$, which requires more time to receive the same amount of information as that of clients $R_1$ and $R_2$. Therefore, some time is wasted for clients $R_1$ and $R_2$ since the information sent by the source is not innovative to these two receivers after they have received the required data. On the other hand, if these two clients have packets that are not received by client $R_3$, one of them can forward the packets to client $R_3$ on behalf of the source on a different channel using its residual time, while the other client can continue receiving information from it. Ideally, if clients $R_1$, $R_2$ are assigned to devote $1/7$ and $1/3$ of their time to serve client $R_3$ alternately while spending the rest of their time to receive from the BS, then the maximum achievable throughput for $R_1$ is $0.7 \times \frac{1}{7} = 0.1$, and $R_2$ is $0.9 \times \frac{1}{3} = 0.3$. Meanwhile, client $R_3$ can receive data alternately from clients $R_1$, $R_2$ when they are active, and from the BS in the rest time, so its achievable throughput is $\frac{1}{7} \times 0.9 + \frac{1}{3} \times 0.8 + (1 - \frac{1}{7} - \frac{1}{3}) \times 0.4 > 0.6$ (see Fig. 3(b)), which suggests that the throughput of 0.6 (packet/slot) can be achieved through this cooperation scheme.

The key to the success of this cooperative strategy is the scheduling of the relay transmissions, that is, to determine which set of relay node should transmit at a specific time slot. To this end, we adopt a stochastic scheduling method, which works as follows. At the beginning of each time slot, the source generates a random variable $x$ uniformly distributed in $[0, 1]$. If $0 \leq x < \frac{1}{6}$, then $R_1$ is selected to relay the data to $R_3$, while $R_2$ keeps receiving from the source. If $\frac{1}{6} \leq x < \frac{1}{3} + \frac{1}{6}$, the roles of $R_1$ and $R_2$ are exchanged. Otherwise, only the source transmits and all clients receive information from it. This scheduling decision is broadcasted to all relays. Each second-hop receiver always receives from the best transmitter (the source or a relay). An example of the scheduling sequences is shown in Fig. 3(c).

The source and the selected relays will transmit at the scheduled time slot. The packets to be transmitted are encoded using the MWN technique, which is well defined for the base station in Section 3. However, for the selected relays, with high probability, they have not decoded all the packets within the current expected window of the base station (ranging from $|V \cdot t| - W + 1$ to $|V \cdot t|$) at time $t$. This case, the relays might not be knowledgeable to generate a symbol from packets in the expected window.

We restate the notion of “heard” for a packet which was introduced in [20]. A packet is said to be “heard” if the receiver knows some linear combination involving that packet. Based on the previously received symbols in the buffer, a selected relay generates a symbol closest to the expected window and transmits it on the relaying channel. For example (see Fig. 3(d)), the batch of packets to be encoded include the decoded packets as well as the “seen” and “heard” packets (e.g., $p_{10}$, $p_{11}$, and $p_{12}$). The relays’ encoding window may lack the expected window for some distance depending on its previous receiving status from the base station. In Section 5, we will see that such information backlog should not be large. Note that all active transmitters use different channels to avoid interference. Each receiver listens to the channel of its best transmitter (either the base station or a relay), and attempts to decode the packets using Gaussian elimination, as explained in Section 3.

4.2. MWNCast protocol

In this subsection, we discuss the details of MWNCast protocol. We have explained how to implement MWN in a cooperative scenario, so in the following we focus on the cooperative scheduling in MWNCast, which can be
decomposed into three modules, namely the selection of relay nodes, the allocation of relay time, and the online scheduling of relay transmissions.

4.2.1. Relay node selection

The capacity (packet/slot) of single-hop broadcast is limited by the bottleneck receiver. Yet it is still unknown the extent to which the system throughput can be increased when the receivers are allowed to cooperate. Note that finding the achievable cooperative throughput is essential for the rate control of MWN. To achieve a high cooperative throughput, we employ a binary search procedure as shown in Algorithm 1. The algorithm maintains a lower threshold $C_l$ and an upper threshold $C_u$ for the target throughput initially. Then starting with $C_T = (C_l + C_u)/2$, a set of relay nodes with qualified link capacities are determined (lines 5–6). The feasibility of the target throughput $C_T$ is checked using Algorithm 2 (line 7), which computes the achievable throughput of the remaining nodes by jointly considering the optimal relay selection and the corresponding relay time allocation (to be discussed). If the target throughput can be achieved by the cooperation among the nodes, then the lower threshold is increased to $C_T$ (line 9), otherwise the upper threshold is reduced to $C_T$ (line 11). The same procedure is repeated until the upper and lower thresholds converge. Finally, the algorithm returns the highest throughput $C^*$, which is known to be achievable.

For a target throughput $C_T$, if a node $i$ has a PRP of $c_{0i} > C$, it is selected as a candidate relay node. In this case, at least a proportion of $C_T/c_{0i}$ of its time has to be used for receiving data from the BS so that the target throughput requirement can be satisfied. Therefore, its residual time is at most $(c_{0i} - C_T)/c_{0i}$, which can be used for serving the remaining receiver nodes.

To maximize the ultimate throughput, in each round of the feasibility check Algorithm 2, a subset $R_i$ from $R$ with at most $K – 1$ elements is selected so that the overall capacity of all nodes in $E$ is maximized (line 7), where the capacity of a node $j$ is determined by $c_{R(i,j)}$, where $R(j,l)$ is the node in $R_i' = R_i \cup \{s\}$ with the maximum capacity to node $j$. This problem is a special case of the generalized maximum coverage problem [30], which is known to be NP-hard. To solve this problem, we introduce the following definitions:

**Definition 2** (residual capacity/weight). Consider a selection $R_i$, a relay $i$ and a receiver $j$. We define the residual capacity $c_{R(i,j)}$ to be equal to $c_{ij} - c_{R(i,j)}$.

**Definition 3** (addition of a relay). For a selection $R_i$ and a relay $i \notin R_i$, we define $R_i \oplus i$ as the addition of $i$ to $R_i$. In other words, $R_i \oplus i$ is a new selection $R_i'$, and

$$R'(j) = \begin{cases} j, & \text{if } c_{ij} > c_{R(i,j)} \\ R(j, l), & \text{otherwise} \end{cases}$$

(2)

Based on these concepts, we develop a greedy algorithm as shown in Algorithm 3. The basic idea is to incrementally add the relay node with the maximum positive residual capacity, so that the overall capacity is non-decreasing.

At line 3, all receivers are initially assigned to the source. Then in each round, one of the candidate relays that has the maximum positive residual capacity is selected to join the relay node set $R_i$ (line 6) until $|R_i|$ exceeds $K$. It can be proved that this greedy algorithm can achieve an approximation ratio of $1 - (1 - 1/K)$ to the optimal solution [30].

Taking Fig. 3(a) as an example, if we check the feasibility of 0.6, clients 1 and 2 have the residual capacity to serve receiver 3. The relays with better channel conditions to the unsatisfied receivers are given a higher priority, e.g., client 1.

**Algorithm 1.** Find the achievable cooperative throughput.

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$C_L \leftarrow 0, C_U \leftarrow 1$;</td>
</tr>
<tr>
<td>2</td>
<td>$C_T \leftarrow (C_L + C_U)/2$;</td>
</tr>
<tr>
<td>3</td>
<td>while $C_U - C_T &gt; \Delta$ do</td>
</tr>
<tr>
<td>4</td>
<td>$C_T \leftarrow (C_T + C_L)/2$;</td>
</tr>
<tr>
<td>5</td>
<td>$R \leftarrow {j: c_{0j} \geq C_T, j \in N}$;</td>
</tr>
<tr>
<td>6</td>
<td>$E \leftarrow N \setminus R$;</td>
</tr>
<tr>
<td>7</td>
<td>Call Algorithm 2 to check the feasibility of $C_T$ for the relay set $R$ and the receiver set $E$;</td>
</tr>
<tr>
<td>8</td>
<td>if $C_T$ is feasible then</td>
</tr>
<tr>
<td>9</td>
<td>$C_L \leftarrow C_T$;</td>
</tr>
<tr>
<td>10</td>
<td>else</td>
</tr>
<tr>
<td>11</td>
<td>$C_U \leftarrow C_T$;</td>
</tr>
<tr>
<td>12</td>
<td>end</td>
</tr>
<tr>
<td>13</td>
<td>end</td>
</tr>
</tbody>
</table>

4.2.2. Relay time allocation

Given the relay subset $R_i$, the next step is to decide the time ratio $\tau_i$ that they can devote for relaying. Note that since none of the relay nodes should contribute more than its residual time, and none of the receiver nodes should get service more than its residual throughput demand, the relay time $\tau_i$ for this subset $R_i$ is set to the minimum of the residual time of these relay nodes and the residual demand of all second-hop receivers (line 8 in Algorithm 2). Then for each selected relay node $i$, its residual time is reduced by the amount of $\phi_i$ (line 10). If its residual time is used up, it is removed from the candidate relay set and will not participate in the relay time allocation in the next round. Similarly, for each receiver $j$, its residual demand is reduced by an amount of $\phi_j = c_{R(i,j)}$, which is the effective throughput it will receive from this set of relay nodes (line 14). If its demand is satisfied, it is removed from receiver set and will not be considered in the next round (line 11). The same procedure is repeated to find the next subset of relay nodes and its relay time allocation, until either the candidate relay set $R$ or the receiver set $E$ becomes empty, or the overall relay time is used up (line 8). If the receiver set $E$ is empty eventually, it means that the target throughput demand $C_T$ can be met by all receiver nodes, then the algorithm returns a list of relay subset and their corresponding relay time;
otherwise, it means the target throughput $C_T$ is infeasible and the algorithm returns an empty set.

4.2.3. Online relay transmission scheduling

From Algorithms 1 and 2, we can find a list $\mathcal{L}$ of relay node set $\mathcal{R}_l$ and the corresponding relay time allocation $\phi_l$, such that the multicast throughput of the system is high and known to be achievable. Since $C$ denotes the ultimate throughput corresponding to the results, for any throughput requirement $C \leq C$, we have:

$$C \leq \sum_{j \in \mathcal{L}} \psi_l \cdot c_{\mathcal{R}_l,j}, \forall j \in \mathcal{N}. \tag{3}$$

From (3), we can see that the amount of time that a subset $\mathcal{R}_l$ to be scheduled for relaying should be proportional to $\frac{1}{C}$, and the corresponding relay time allocation is scheduled for $\frac{1}{C}$.

For this relay is scheduled for $\frac{1}{C}$, and receiver as to successfully receive the packet from the best relay $\frac{1}{C}$ and $\frac{1}{C}$ for node $j$.

Algorithm 2. Check the feasibility for $C_T$.

```plaintext
1 begin
2 $C_l \leftarrow (c_{ij} - C_l) / c_{0l}, \forall i \in \mathcal{L};$
3 $D_j \leftarrow C_l, \forall j \in \mathcal{E};$
4 $I \leftarrow 0;$
5 while $\mathcal{R} \neq 0$ and $\mathcal{E} \neq 0$ and $\sum_i \phi_i \leq 1$ do
6 \hspace{1em} $I \leftarrow I + 1;$
7 \hspace{1em} Call Algorithm 3 to select $\mathcal{R}_l \subseteq \mathcal{R} \setminus \{s\}$ such that $|\mathcal{R}_l| \leq K - 1$ and $\sum_{j \in \mathcal{E}} c_{\mathcal{R}_l,j}$ is maximized, where
8 \hspace{1em} $R(j, I) \leftarrow \arg \max_{i \in \mathcal{R} \setminus \{s\}} \frac{a_j}{c_{ij,l}}$
9 \hspace{1em} $\phi_l \leftarrow \min_{i \in \mathcal{R}_l} C_l, \min_{j \in \mathcal{E}} D_j / c_{\mathcal{R}_l,j}, 1 - \sum_i \phi_i;$
10 \hspace{1em} foreach $i \in \mathcal{R}_l$ do
11 \hspace{2em} $C_i \leftarrow C_i - \phi_i;$
12 \hspace{2em} if $C_i \leq 0$ then $\mathcal{R} \leftarrow \mathcal{R} \setminus \{i\};$
13 \hspace{1em} endforeach
14 \hspace{1em} foreach $j \in \mathcal{E}$ do
15 \hspace{2em} $D_j \leftarrow D_j - \phi_i * c_{\mathcal{R}_l,j};$
16 \hspace{2em} if $D_j \leq 0$ then $\mathcal{E} \leftarrow \mathcal{E} \setminus \{j\};$
17 \hspace{1em} endforeach
18 if $\mathcal{E} = \emptyset$ then return $(\mathcal{R}_l, \phi_l)_{\mathcal{L}, \mathcal{E}};$
19 else return $\emptyset;$
20 end
```

In each time slot $t$, the source generates a random number between 0 and 1, if its value falls between $\psi_j$ and $\psi_{k+1}$, then the $k$th subset of relay nodes are selected for relaying in this time slot. It is easy to see that this stochastic scheduling policy converges to the required proportion of time for each relay set in the long run. This schedule algorithm can be executed by the source in an online fashion at the beginning of each time slot, and a unique channel is assigned to each selected relay node. The scheduling results (relay nodes and their operating channels) are broadcasted to all receivers, then they can choose the best relay node and switch to the corresponding channel to receive the data.

5. Analysis

In this section, we develop some theoretical models to characterize the basic properties of MWNCast. Firstly, we introduce the equivalent channel capacity model, which is an unified model for characterizing the capacity of both relay and receiver nodes. Based on this model, the decoding delay, reliability and decoding complexity properties of MWNCast are analyzed using the random walk and point process theories.


```plaintext
1 begin
2 $\mathcal{R}_l \leftarrow \{s\};$
3 $R(j, I) \leftarrow s, \forall j \in \mathcal{E};$
4 while $|\mathcal{R}_l| \leq K$ do
5 \hspace{1em} Find a relay $i \in \mathcal{R}$ with the maximum residual capacity, i.e.,
6 \hspace{2em} $i \leftarrow \arg \max_{j \in \mathcal{R}_l} \sum_{j \in \mathcal{E}} c_{\mathcal{R}_l,j};$
7 \hspace{2em} if $c_{\mathcal{R}_l,j} > 0$ then
8 \hspace{3em} $\mathcal{R}_l \leftarrow \mathcal{R}_l \cup \{i\};$
9 \hspace{2em} else break;
10 end
11 return $\mathcal{R}_l.$
12 end
```

5.1. Equivalent channel capacity model

As discussed in Section 2, the capacity of a point-to-point wireless link $(i, j)$ is given by the PRP $c_{ij}$. However, the analysis of the link capacity in MWNCast is complicated since: (i) a relay node may not stay in the “receiving” state all the time; (ii) a receiver node may receive data from different relay nodes at different time slots. In this subsection, we propose an equivalent channel capacity model to characterize the capacity of these two kinds of nodes.

For a relay node $i \in \mathcal{R}$, its aggregated fraction of time in the “relaying” state is given by $\Phi_i = \sum_{E \in L} c_{\mathcal{R}_l,j}$. Since the online relay scheduling algorithm is i.i.d. across time slots, in each time slot, the probability for the node to receive from the source is $1 - \Phi_i$, and the probability for relaying is $\Phi_i$. Taking the PRF from the source into account, we can define the equivalent channel capacity $C_i$ for this relay node as $C_i = (1 - \Phi_i) c_{0l,j}$, which is the maximum achievable throughput of this node from the source without errors.

For a receiver node $j$, if a relay subset $\mathcal{R}_l$ is scheduled for transmission (with a probability of $\Phi_l$), it will choose the best relay node $R(j, I)$ (including the source) with the maximum PRP to receive information, i.e., $R(j, I) = \arg \max_{j \in \mathcal{R}_l} \sum_{j \in \mathcal{E}} c_{\mathcal{R}_l,j}$. Since the stochastic scheduling is i.i.d. across time slots, in any time slot, a relay set $\mathcal{R}_l$ is scheduled with probability $\Phi_l$ and receiver $j$ has probability $c_{\mathcal{R}_l,j}$ to successfully receive the packet from the best relay node in $\mathcal{R}_l$. Therefore, summing over all possibilities, we can define the equivalent channel capacity $C_j$ for node $j$. 


as the probability of successful packet reception in one time slot, i.e., $\hat{C}_t = \sum_{i,C} P(\hat{C}(i,t)).$

Note that this equivalent channel capacity model is an approximation of the link capacity for the two kinds of nodes in MWNCast, which makes it tractable to analyze the reliability and decoding delay properties in the following subsections.

### 5.2. Preliminary property of MWNC

In this subsection, we establish some basic properties for MWNC using the random walk and point process theories, with which we can analyze the performance of MWNCast.

For any receiver with capacity $\hat{C}$, let us define $I(t)$ as the total number of received innovative symbols up to time $t$, $G(t)$ as the number of packets which are inevitably lost up to time $t$. Note that not all innovative symbols can be used for decoding the original packets. For example, in Table 1, the symbol received at $t = 8$ becomes useless and shall be discarded at the end of time $t = 12$ since $p_7$ cannot be decoded ever since. We define $D(t)$ as the number of discarded symbols up to time $t$. Then $I(t) - D(t)$ represents the number of received innovative symbols which contain the information for the packets covered by the coding window except for the lost $G(t)$ packets. For better understanding, we list all the variables in Table 2 corresponding to the example shown in Table 1.

Similar to [20], it is assumed that the field size is sufficiently large and if packet $p_i$ has been “seen” by a receiver, successful reception of a new coded symbol which involves packet $p_{i+1}$ will let the receiver “see” the packet $p_{i+1}$. This can be observed from the instance shown in Fig. 2(a). The reception of $s_7$ allows the receiver to “see” $p_8$, since $p_7$ has already been “seen” and decoded. The subsequent receptions of $s_9, s_{10}$ let the receiver “see” $p_{11}, p_{12}, p_{13}$ respectively.

For a MWNC’s receiver with capacity $\hat{C}$, let us define a random process which is denoted by a particle in $\mathcal{R}^1$ with its value at time $t$ given by

$$S(t) = V \times t - G(t) - (I(t) - D(t)).$$

We have the following results regarding the decoding and loss events for MWNC.

**Lemma 1.** Decoding event occurs at time $t$ if and only if $S(t) \leq 0$ at the end of this time slot. All the packets from the last foremost decoded (or lost) packet to the head of current window will be decoded.

**Proof.** Decoding event occurs at the moment when the coding coefficient matrix in buffer is full rank. In this case, the number of innovative symbols in buffer must be as many as the number of packets covered by the window by time $t$ except for the lost ones, i.e., $|V \times t| - G(t)$. Therefore, we have $(I(t) - D(t)) = |V \times t| - G(t)$, i.e., $S(t) \leq 0$. $\square$

**Lemma 2.** Recovery loss event occurs at time $t$ if and only if $S(t) > |V \times t| - W$ at the end of $t$ event. Moreover, all the un-decoded packets before the tail of the window are lost.

**Proof.** First, we will show that recovery loss is inevitable if and only if the packet $|V \times (t + 1)| - W$ has not been “seen” at the end of slot $t$. Notice that the last packet in the coding window moves to $|V \times (t + 1)| - W + 1$ at time $t + 1$. So if the packet right before $|V \times (t + 1)| - W$ has not been “seen” (i.e., a symbol contains this packet has not been received before), then all the un-decoded packets before the window will be impossible to get decoded. On the hand, if the packet $|V \times (t + 1)| - W$ has been “seen”, it is still possible to decode the packet $|V \times (t + 1)| - W$ and no new packet loss occurs in slot $t$.

Second, we show that the index of foremost “seen” packet is $G(t) + (I(t) - D(t))$. Since there are $I(t)$ innovative symbols received, among which a number of $D(t)$ symbols have been determined as useless and discarded, and there
are a number of $G(t)$ packets which have been lost before, the index of the foremost “seen” packet is $G(t) + (I(t) - D(t))$.

Therefore, recovery loss is inevitable if and only if, $[V \times (t + 1)] - W > G(t) + (I(t) - D(t))$, which is equivalent to $S(t) > W - V$ by Eq. (5).

From Table 2, we can see that in the 12th slot, $S(t) = 3$ which makes the receiver detect a recovery loss. As will be discussed, whenever recovery loss occurs, the particle will be immediately bounced back for a distance of 1.

**Lemma 3.** If a new packet is lost in time $t$, the number of newly discarded symbols in buffer must be exactly one less than the number of packets just got lost, i.e.,

$$\Delta D(t) = \Delta G(t) - 1.$$  

**Proof.** As shown in the proof of Lemma 2, when new recovery loss occurs in slot $t$, the packet $[V \times (t + 1)] - W$ has not been “seen”. Since each received symbol let the receiver “see” one packet, this implies that the number of packets newly lost must be at least one more than the number of symbols newly discarded, i.e., $\Delta D(t) \geq \Delta G(t) - 1$.

To prove the other direction, suppose the packet $[V \times (t + 1)] - W - 1$ has not been “seen” and is determined as lost in slot $t$. Because the coding window moves by at most one packet in one slot, by Lemma 2 the packet $[V \times (t + 1)] - W - 1$ should have been determined as lost in slot $t - 1$. This leads to the contradiction with the assumption that the packet $[V \times (t + 1)] - W - 1$ is newly lost.

Accordingly, for the specified receiver, MWNC can be modeled as a one-dimensional random walks [31] (see Fig. 4). The random walk has two reflecting barriers at $-V$ and $W - V$ corresponding to the decoding and recovery loss events respectively, where $S(t)$ in (5) corresponds to the position of the random walk at time $t$. Let $X$ denote the step size of the random walk, which is a random variable with the following density function:

$$f(x) = \frac{C}{d_d} \delta(x + d_d) + (1 - C) \delta(x - d_b),$$  

where $d_d = 1 - V$ and $d_b = V$. Moreover, the mean and variance of a step are denoted as $\mu = V - C$ and $\sigma^2 = Var(X) = C(1 - C)$.

The following proposition specifies the behavior of the random walk representing the specified receiver:

**Theorem 1.** At time $t$, if the particle crosses the left barrier $-V$, it will be reflected rightward for a distance of $d_b$ in the next time slot. If the particle crosses the right barrier $W - V$, it will be immediately bounced back for a distance of 1. Otherwise, the particle will make a random move according to (7).

**Proof.** Firstly, notice that if and only if $S(t - 1) > -V$, the received symbol at time $t$ is innovative. That is, the window’s foremost packet $[V \times t]$ is informative to the receiver since $V(t - 1) - G(t - 1) - (I(t - 1) - D(t - 1)) > -V$.

Therefore, when the particle does not cross the two barriers, its position at time $t$ relative to the last slot $S(t) = S(t - 1) + V - \Delta t(t)$ depends on whether a symbol is received successfully, which follows the step function defined in (7). If the particle just crosses the left barrier $-V$, the received symbol contains no new information. Thus $I(t) = I(t - 1)$ and the particle moves rightward definitely. If the particle crosses the right barrier $W - V$, it is indicated in Lemma 3 that it will be bounced back instantly by a distance of 1.

5.3. Reliability analysis

The reliability analysis is complicated in MWNCast since the second-hop receivers might suffer from larger recovery loss ratio. However, it is easy to see that the proposed stochastic scheduling policy converges to the required proportional of time for each relay set in a long run and consequently the information backlog between the relays and the base station should not be large. In addition, the symbols to be transmitted by the relays are generated randomly, so that the probability that they contain innovative information to the second-hop receivers is greatly increased. Therefore, without differentiating the relays and receivers, we assume a specified node receives MWNC’s encoded symbols of rate $V$ packet/slot from the information source on an equivalent channel of capacity $C$ packet/slot. Based on this assumption, the recovery loss ratio for both the relays and the receivers can be derived. In Section 6, we will see that the simulation results match well with the theoretical results.

To evaluate the reliability of MWNCast at one receiver, we are concerned about only two statistics, the number of packets which are decoded, and the number of packets which cannot be recovered and get lost. Let “D” and “L” represent the events of new packet “Decoded” and new packet “Lost” at the receiver, respectively. Remind that MWNCast can be modeled by a random walk as shown in Fig. 4, we are able to characterize the packet decoding and packet loss events in MWNCast by a two-state point process (see Chapter 9 in [31]), as shown in Fig. 5. Specifi-
ally, if the “Decode” (D) event occurs at some time, with probability \( p_{DD} \) it will return the same state after a random time interval \( T_{DD} \), and with probability \( p_{DL} = 1 - p_{DD} \) it will make a transition to the “Loss” (L) state after a time interval \( T_{DL} \). Similarly, we can define \( p_{LD} \), \( p_{LD} \) and \( T_{L} \) as the transition probabilities and transition time for the “Loss” state. These quantities can be derived using the random walk and point process theories as follows.

Firstly, let us define \( G(\theta) \) as the moment generating function of \( X \), which is the two-sided Laplace transform of the step function \( f(x) \) defined in (7), that is,

\[
G(\theta) = E[e^{-\theta X}] = \hat{C}e^{-\theta(1-V)} + (1 - \hat{C})e^{\theta V}. \tag{8}
\]

From the property of moment generating function, we know that: (i) \( G(\theta) \) is a convex function; (ii) If \( E[X] \neq 0 \), there are two roots for the equation \( G(\theta) - 1 \), one is \( \theta = 0 \), the other is \( \theta = 0.9 \), who has the same sign as \( \mu \).

Let \( -B \) and \( A (A, B > 0) \) denote two absorbing barriers for the random walk starting at the origin, we can define the stopping time \( N \) as

\[
N = \min\{n : S(n) \leq -B \text{ or } S(n) \geq A\}, \tag{9}
\]

which is the number of steps to cross one of the barriers starting from the origin.

We can define the moment generating function \( G_N(\theta) \) with respect to \( N \) and \( S(N) \), that is,

\[
G_N(\theta) = E[e^{-\theta N(S)}]. \tag{10}
\]

Suppose that we set \( s = G(\theta)^{-1} \), then we have \( G_N(\theta) = E[e^{-\theta N(S)} G(\theta)^{-1}] \). For this equation, we can find \( \theta = \theta_0 \) such that \( G(\theta) = 1 \), then it is easy to verify that \( e^{-\theta_0 N(S)}G(\theta)^{-1} \) is a martingale with mean 1 since it is the product of independent unit mean random variables. According to the martingale stopping theorem (Theorem 6.2.2 in [32]), we can obtain:

\[
E[e^{-\theta_0 N(S)}] = 1. \tag{11}
\]

Since the events of \( S(N) \leq -B \) and \( S(N) \geq A \) are independent, from (9) and (11), we have,

\[
E[e^{-\theta_0 N(S)} | S(N) \geq A] p_A + E[e^{-\theta_0 N(S)} | S(N) \leq -B] p_{-B} = 1. \tag{12}
\]

For the absorbing states \( A \) and \( -B \), we can get the following approximations:

\[
E[e^{-\theta_0 N(S)} | S(N) \geq A] \approx e^{-\theta_0 A}, \quad E[e^{-\theta_0 N(S)} | S(N) \leq -B] \approx e^{\theta_0 B}.
\]

Substituting these two approximation equations into (11), and using the fact that \( p_A + p_{-B} = 1 \), we can get the probabilities of absorption at \( A \) and \( -B \) as

\[
p_A \approx \frac{1 - e^{\theta_0 B}}{e^{-\theta_0 A} - e^{\theta_0 B}}, \quad p_{-B} \approx -\frac{1 + e^{-\theta_0 A}}{e^{-\theta_0 A} - e^{\theta_0 B}}. \tag{13}
\]

where \( \theta_0 \) is the non-zero root of the equation \( G(\theta) = 1 \).

To derive the distribution for \( N \), let \( \lambda_1(s) \) and \( \lambda_2(s) \) denote two real roots of the equation \( G(\theta) = 1/s \). Then from (10), we can obtain a different expression of (11) with respect to \( N \):

\[
E[e^{-\theta_1(s) N(S)}] = 1, \quad E[e^{-\theta_2(s) N(S)}] = 1. \tag{14}
\]

Using the approximation \( S(N) \approx A \) when \( S(N) \geq A \), and \( S(N) \approx -B \) when \( S(N) \leq -B \), we have:

\[
p_A e^{-\theta_1(s) E_A(s^N)} + p_{-B} e^{-\theta_1(s) E_{-B}(s^N)} = 1, i = 1, 2. \tag{15}
\]

where \( E_A(s^N) \) and \( E_{-B}(s^N) \) denote the conditional expectations at \( A \) and \( -B \), respectively. Using \( p_A \) and \( p_B \) given by (13), we can obtain \( E_A(s^N) \) and \( E_{-B}(s^N) \) from (15). Then we have the moment generating function for \( N \) as:

\[
E[S^N] = p_A E_A(s^N) + p_{-B} E_{-B}(s^N). \tag{16}
\]

By differentiating Eq. (16) with respect to \( s \), we can obtain the first and second moments of \( N \) respectively.

To derive the recovery loss ratio, we assume the particle is always located at the largest possible position after an event, which gives the upper bound for the loss probability. By Theorem 1, after an “D” event, the particle’s maximum position is at \( d_k \). After a “L” event, the particle must get back to at most \( W - 1 \) before making a random move. Therefore, the transition probabilities \( p_{DD}, p_{DL}, p_{LD} \) and \( p_{ID} \) between these two events, and the expected transition time \( T_{DD}, T_{DL}, T_{LD} \) and \( T_{DL} \) can be derived using (13) and (16) respectively assuming the particle starting from the corresponding position.

The equilibrium distribution of the embedded Markov chain for two-state point process can be given by \( (\pi_D, \pi_L) = \left( \frac{p_D}{\pi_D + p_D}, \frac{p_L}{\pi_L + p_L} \right) \). Let us define \( T = \pi_D T_{DD} + \pi_L T_{DL} + \pi_D T_{DL} + \pi_L T_{LD} \), then the proportion of time passed from states “L” and “D” to “L” are given by \( \pi_D T_{DL}/T \) and \( \pi_L T_{DL}/T \), respectively.

Note that MWNC has two recovery loss scenarios depending on the previous event, “L” to “L” and “D” to “L”. If it is from “L” to “L”, all the packets covered by the window will get lost, so the number of lost packets should be proportional to the transition time \( T_{DL} \). If it is from “D” to “L”, then the lost packets should be the number of packets covered from the time of previous event minus \( W \) since only the packets behind the window will get lost. According to the theory of point process (see page 356 [31]), the overall recovery loss probability can be obtained as:

\[
P_{\text{loss}} = \frac{\pi_D p_{DL}(T_{DL} - W/V) + \pi_L p_{LD} T_{L}}{T}. \tag{17}
\]

### 5.4 Decoding delay analysis

For a receiver in MWNCast, the delay for receiving a packet is composed of two parts: the queuing delay and the decoding delay. The queuing delay can be analyzed using the similar procedure in [18]. In the following, we will focus on the decoding delay and consider a saturated system in which the source always has packets to transmitted.

Attributed to the stochastic scheduling policy, any client (a relay or a receiver) in the network can be approximately
considered as connected to the information source by MWNC on an equivalent channel of capacity \( \mathcal{C} \). Since the further the particle is away from the “decoding” barrier, the longer time on average the corresponding receiver will have to wait before the next decoding event occurs. In this case, we derive an upper bound for the average decoding delay assuming that the right barrier moves to infinity, whereby the random walk is simplified to a single left barrier at \( 0 \) with a starting point \( d_0 \). Note that this is also the case that \( W \to \infty \), when recovery loss can be negligible as indicated by the last subsection. One little pitfall is that how can decoding delay be finite if \( W \to \infty \)? One extreme example \( (W = 1000, V = \mathcal{C} = 1) \) will make this clear. In the first slot, by definition of MWNC, instead of sending a random combination of \( p_1, \ldots, p_{1000} \), the source sends \( p_1 \) and the receiver, upon receiving the symbol \( (\mathcal{C} = 1) \), will decode \( p_1 \) instantly. It is obvious that the receiver will decode one new packet from the newly received coded symbol in every subsequent time slot even if \( W = 1000 \).

The barrier at \( 0 \) indicates the moment of decoding. Assume the particle always reflects back to the largest possible position \( d_0 \) after a decoding event. According to Eq. (16), when \( A \) approaches to infinity and \( B \) is set to \( d_0 \), we have \( P_{\geq 1} = 1 \) and \( P_{= 0} = 0 \). The first two moments of \( N \) are derived from (16) as follows:

\[
E[N] \simeq -\frac{d_0}{\mu}, \quad E[N^2] \simeq \frac{d_0^2 \mu - \sigma^2 d_0}{\mu^3}.
\]

The decoding delay for a packet is defined as the time duration from the moment that it is encoded to the time that it is decoded. We can model the decoding process as a renewal process. Then the sum of decoding delay for the packet in a renewal period \( N_i \) is given by

\[
D(N_i) \simeq \frac{N_i}{2} N_i V, 
\]

where \( N_i/2 \) is the average decoding delay for a packet, \( N_i V \) is the average number symbols transmitted during this time period.

By the theory of renewal reward process and the definition of average decoding delay, we have

\[
D = \lim_{t \to \infty} \frac{1}{t} \sum_{i=1}^{\infty} D(N_i) \leq \frac{E(D)}{E(N)} \cdot V \frac{1}{2} E[N^2] = \frac{1}{2} E[N^2].
\]

Let \( \rho = V/\mathcal{C} \) denote the traffic intensity of the receiver, then from (7), (18) and (19), we have

\[
D \leq \frac{1}{2} \left( \frac{1}{\rho} - \frac{1}{1 - \rho} \right)^2 + \frac{\rho}{1 - \rho}. \quad (20)
\]

Therefore, an upper bound of the decoding delay for the particular receiver is \( O\left(\frac{1}{(1-\rho)^2}\right) \) asymptotically.

5.5. Decoding complexity

In this part, we analyze the decoding complexity of MWNC assuming the window size \( W \) is sufficiently large such that the recovery loss can be negligible.

The decoding complexity of MWNC is composed of two parts: the forward elimination and the backward substitution. Suppose that a receiver has just decoded all packets up to \( [V \times t_0] \) at time \( t_0 \) and \( t_1, t_2, \ldots, t_{K-1} \) denote the time instances that the receiver receives a set of \( k - 1 \) encoded symbols but cannot decode them, until at time \( t_k \) it receives the \( k \)th symbol and is able to decode all the received symbols. In the following, we use \( s_j \) to denote the symbol received at time \( t_j(j = 1, \ldots, k) \).

Lemma 4. The number of nonzero entries in the jth symbol \( s_j \) after the forward elimination is \( |S(t_j)| + 1 \).

Proof. The forward elimination can assure that all the previously “seen” packets can be reduced from the newly received symbol. At time \( t_j \), the receiver has received \( I(t_j) - D(t_j) \) useful symbols. Hence, except for \( G(t_j) \) inevitably lost ones, all the packets up to \( G(t_j) + I(t_j) - D(t_j) \) have been “seen” even they may not be all decoded. The new symbol generated in the \( t_j \)’s coding window can be reduced at the corresponding positions except for the last one \( G(t_j) + I(t_j) - D(t_j) \), which becomes “seen” for the sake of the symbol received at \( t_j \). Hence, there are \( |V \times t_j| - G(t_j) - (I(t_j) - D(t_j)) + 1 \) nonzero entries are left, which equals to \( |S(t_j)| \) according to Eq. (5). \( \Box \)

Lemma 5. The number of arithmetic operations\(^7\) in finite field required for forward elimination of the jth symbol \( s_j \) is

\[
W - |S(t_j)| - j + \sum_{i=1}^{j-1}(|S(t_i)| + 1) \text{ when } j + |S(t_j)| < W, \quad \text{and } \sum_{i=1}^{j-1}W + 1 + |S(t_j)|(|S(t_j)| + 1) \text{ otherwise.}
\]

Proof. The packets with indexes from \( [t_j \times V] - W + 1 \) to \( [t_j \times V] \) are used to generate the specific symbol. Among them, the receiver may have decoded a number of packets. Since as Lemma 4 suggests, the number of nonzero entries after elimination is \( |S(t_j)| + 1 \) and there are \( j - 1 \) previously eliminated results, the receiver only has a decoded intersection with the coding window if \( j + |S(t_j)| < W \). In this case, the first \( W - |S(t_j)| - j \) packets in the window are already decoded by the receiver, so the same number of operations and are needed to eliminate these entries. Then, the previously reduced symbols \( s_i, i < j \) can be used to eliminate the corresponding entries, which takes \( |S(t_i)| + 1 \) operations for each symbol \( s_i \) according to Lemma 4. When \( j + |S(t_j)| \geq W \), there are no decoded packets in the coding window, thus the receiver can only use the existing eliminated results to reduce the symbol. The elimination for the \( W - |S(t_j)| - 1 \) positions takes \( \sum_{i=j}^{j-1}W + 1 + |S(t_j)|(|S(t_j)| + 1) \) calculations. \( \Box \)

Theorem 2. Given a given achievable throughput \( V \) packet/slot, MWNC can achieve the optimal decoding complexity of \( O(W) \).

Proof. A network coding symbol is encoded with \( W \) packets, a receiver needs at least \( W - 1 \) operation to decode the original information, so a trivial lower bound for the decoding complexity is \( O(W) \). Therefore, it is sufficient to prove that, for every receiver in the network, the decoding complexity of MWNC is upper bounded by \( O(W) \).

\(^7\) We take one time of multiplication and addition as one operation.
According to Lemma 5, an obvious upper bound of complexity for the jth symbol’s elimination procedure is
\[ W + \sum_{i=1}^{j-1} |S(t_i)| + 1. \]
Thus, the total number of computations for the forward elimination of k packets is upper bounded by
\[ \sum_{j=1}^{k} (W + \sum_{i=1}^{j-1} |S(t_i)| + 1). \]
The total number of computations for backward substitution is
\[ \sum_{j=1}^{k} |S(t_i)| + 1, \]
so the overall complexity \( \Omega(k) \) for decoding k packets is bounded by:
\[
\Omega(k) \leq Wk + \frac{(k+1)k}{2} + \sum_{j=1}^{k} \sum_{i=1}^{j-1} |S(t_i)| \leq \frac{(k+3)k}{2} W + \frac{(k+1)k}{2},
\tag{21}
\]
where the second inequality is valid because \( |S(t_i)| \leq W \).

Note that as discussed in previous subsection, the decoding event occurs after a random time duration of \( N \), so \( k = |V(t + N)| - |V_t| \approx VN \). According to the renewal reward process and (21), the average decoding complexity can be obtained as follows:
\[
\Omega = E(\Omega(N)) \leq \frac{E(N^2)V + 3E(N)}{2E(N)} W + \frac{E(N^2)V + E(N)}{2E(N)}. \tag{22}
\]
From (18), we know that given the throughput, \( E(N) \) and \( E(N^2) \) are independent of \( W \), therefore, the decoding complexity is dominated by the window size, which is on the order of \( O(W) \) from (22).

6. Simulation results

In this section, we provide extensive simulation results to compare the performance of MWNCast with other network coding-based multicast schemes, and also to illustrate the advantages of the MWNC technique. The network topologies in simulations are generated randomly, whereby the location of all clients is uniformly distributed around the source. The channel between any two nodes is assumed to follow the Rayleigh fading channel model. The coefficients of network coding are generated on a \( G(2^8) \) Galois Field. The time duration for each simulation is \( 10^5 \) time slots.

6.1. Throughput and decoding delay

In Fig. 6, we compare the achievable throughput of MWNCast with those of RLNC and ANC under different network sizes. Note that a larger window size will lead to higher encoding and decoding complexity. To avoid unacceptable computation complexity, a window size around 20 is typically employed in many real implementations of RLNC, e.g., [15]. For fair comparison, the window sizes of both MWNCast and RLNC are set to 20. When the channel number \( K = 1 \), MWNCast reduces to MWNC (simple multicast without cooperation). We can see that the achieved throughput of MWNC is close to ANC (which is known to be throughput-optimal), but it outperforms RLNC under all network conditions. It also can be seen that the throughput of RLNC decreases as the network size increases, which has been discussed in the literature [14]. When multiple channels are available, we provide the simulation for RLNC with the same relay scheduling strategy as in MWNCast for fair comparison. It is observed that for \( K = 2 \) and \( K = 3 \), MWNCast preserves its superiority over cooperative RLNC (denoted by CoopRLNC in Fig. 6) with the average performance gain of 21.5% and 22.5%, respectively. Therefore, we can see that by taking advantage of moving window network coding and cooperation, MWNCast is effective in improving the system throughput.

In Fig. 7, we study the tradeoff between the decoding delay and throughput for a network with 100 nodes. Note that the result of ANC is not included since the decoding delay of the receivers with poorer channel condition increases with the simulation time, which is unfair for comparison. In this figure, the lines and bars show the average and maximum decoding delays of all receivers respectively for each scheme. It is noteworthy that, to maintain a given throughput, with the newly proposed MWNC-based scheme, the average decoding delay for a
successful decoded packet is much lower than that with the RLNC-based scheme. The reason is that, without any prior knowledge about the receiving status, the decoding opportunity with MWNC exists in each time slot, but with RLNC, it requires at least \( W \) slots to decode a full block of packets. We also find that the second-hop receiver does not necessarily have larger average decoding delay than the first-hop receiver if the former’s equivalent channel capacity is larger than the equivalent capacity of the latter.

The impact of the window size \( W \) on the average decoding delay of MWNC is shown in Fig. 8. One can observe that under different load conditions, the average decoding delay with a small \( W \) is always lower than that with a large \( W \). This is because when the average decoding delay is measured, only the recovered packets are counted, and a larger \( W \) endows the packets higher probability to be recovered but the average decoding delay will get worse by the packets which may not get recovered otherwise. This phenomenon will be further discussed in next subsection. Finally, the average decoding delay of MWNC is upper bounded by the analytical results which assume that \( W \) approaches infinity and recovery loss can be negligible.

6.2. Reliability

In Fig. 9, we study the effect of window size \( W \) on the packet’s recovery loss ratios of MWNCast under different system traffic load\(^{10}\) conditions when the channel number \( K = 2 \). The recovery loss counted in the simulation can occur at any client in the network. We also calculate the theoretical recovery loss ratio for each receiver in the network and get average as the theoretical recovery loss of the network. From the figure, it can be seen that the theoretical results match well with the simulation results. The congruity between theory and simulation validates the effectiveness of the equivalent channel model. Also, we notice that, the packet’s recovery loss probability drops almost exponentially with the increase of window size. Moreover, we can see the requirement of recovery loss probability of \( 10^{-3} \) can be satisfied with \( W = 20 \) even when the traffic load is as high as 0.9. This manifests an attractive property of MWNC that the increase of the computational complexity is moderate for enhancing the reliability.

If a certain degree of packet recovery loss can be tolerated (i.e. \( 10^{-3} \sim 10^{-1} \)), we study the relationship between the average decoding delay and packet’s recovery loss probability by adjusting the window size. We consider a specific receiver with the packet erasure probability equals to 0.3. As shown in Fig. 10, for a given traffic load, the packet’s recovery loss probability increases with the decrease of window size, but the average decoding delay for the packets not lost also gets smaller. As discussed in Lemma 2, this is because packet recovery loss is inevitable when and only when the last unseen packet is just the one before the lastest packet in the window. Such packets, if not lost, will encumber the decoding of the following packets, leading to larger overall decoding delay. Therefore, with a smaller coding window size, MWNCast acts as a delay filter to force the receiver to drop the packets which may lead to the degradation of the overall delay performance.

---

\(^{10}\) System traffic load is defined as \( \rho = \frac{V}{C} \).
exponentially with the increase of window size. Recovery loss probability of the receivers drops almost reaching the performance of the proposed schemes with the existing solutions.

References


Fei Wu received his B.Sc. and Master degree in Information Engineering from Zhejiang University, China, in 2010 and 2012 respectively. He is currently studying for Ph.D. degree in the Department of Electrical Engineering, the Ohio State University, USA. His research is related to the network coding technique in wireless networks.

Cunqing Hua received the B.Eng. degree in Electronics Engineering from University of Science and Technology of China in 2000. He earned his MPhil. and Ph.D. degree in Information Engineering from The Chinese University of Hong Kong in 2002 and 2006 respectively. He is currently an Associate Professor with the School of Information Security, Shanghai Jiao Tong University, PR China. Prior to that, he was a postdoctoral research fellow at University of Houston (September 2006–December 2008), and served the department of Information Science and Electronic Engineering, Zhejiang University, PR China as an Associate Professor (December 2008–June 2011). He has published over 30 papers in the referred journals and international conferences, in the general area of wireless sensor networks, wireless ad hoc and mesh networks, robust cross-layer resource allocation, cooperative communications, and network coding techniques. He is currently an Editorial Board Member of the International Journal of Computer Networks (IJCN).

Hangguan Shan (M’10) received his B.Sc. and Ph.D. degrees respectively from Zhejiang University and Fudan University, in 2004 and 2009, all in electrical engineering. He was a postdoctoral research fellow in University of Waterloo from 2009 to 2010. In February 2011, he joined the Department of Information Science and Electronic Engineering, Zhejiang University, as an Assistant Professor. He is a co-recipient of the Best Industry Paper Award from IEEE Wireless Communications and Networking Conference (WCNC) 2011, Quintana-Roo, Mexico. His current research focuses on resource management and QoS provisioning in vehicular ad hoc networks, wireless body area networks, and cooperative networks. Dr. Shan has served on the Technical Program Committee (TPC) as member in IEEE VTC-Fall 2010, IEEE VTC-Fall 2011, IEEE iCOST 2011, IEEE IWCMC 2011, and IEEE ICC 2011. He has also served as the Publicity Co-Chair for the third and fourth IEEE International Workshops on Wireless Sensor, Actuator and Robot Networks (WiSARN).

Aiping Huang is a full professor and director of Institute of Information and Communication Engineering at Zhejiang University, China. She was a research scientist at Helsinki University of Technology, Finland from 1994 to 1998. From 1982 to 1994, she was with Zhejiang University, China, as an assistant professor and then an associate professor in the Department of Scientific Instrumentation. She worked from 1977 to 1980 as an engineer at Design and Research Institute of Post and Telecommunication Ministry, China. She graduated from Nanjing Institute of Post and Telecommunications, China in 1977, received MS degree in Communication and Electronic System from Southeast University, China in 1982, and received Licentiate of Technology degree from Helsinki University of Technology, Finland in 1997. She published a book and more than 100 papers in refereed journals and conferences on signal processing, communications and networks. Her current research interests include mobile communications, broadband wireless access, and wireless networks. She is a senior member of IEEE and Chinese Institute of Electronics. She currently serves as a vice chair of IEEE Communications Society Nanjing Chapter, council member of China Institute of Communications, and member of Standing Committee of Zhejiang Provincial Institute of Communications. She is an adjunct professor of Nanjing University of Posts and Telecommunications, China.