Network Coding for Efficient Multicast Routing in Wireless Ad-hoc Networks

Jingyao Zhang, Pingyi Fan, Member, IEEE, and Khaled Ben Letaief, Fellow, IEEE

Abstract—Network coding is a powerful coding technique that has been proved to be very effective in achieving the maximum multicast capacity. It is especially suited for new emerging networks such as ad-hoc and sensor networks. In this work, we investigate the multicast routing problem based on network coding and put forward a practical algorithm to obtain the maximum flow multicast routes in ad-hoc networks. The “Conflict Phenomenon” that occurs in undirected graphs will also be discussed. Given the developed routing algorithm, we will present the condition for a node to be an encoding node along with a corresponding capacity allocation scheme. We will also analyze the statistical characteristics of encoding nodes and maximum flow in ad-hoc networks based on random graph theory.

Index Terms—Network coding, multicast, encoding nodes, Ad-hoc networks, random graph.

I. INTRODUCTION

In recent years, network coding has been attracting more and more attentions. This concept was first proposed by Ahlswede et al. for multicast services in 2000 [1]. They also discussed the solvability of a multicast problem based on the maximum-flow minimum-cut theorem in graph theory. In traditional IP multicast networks, the relay nodes can only replicate and forward the data according to the selected routing, regardless of the content correlation between the data for different destinations. The multicast routes for different destinations may intersect at some nodes and share one common link as their output link. In this case, it is inevitable to employ queuing to the different data packets, which will result in the reduction of the transmission efficiency. Therefore, the maximum flow value cannot be achieved. Moreover, it also makes the routing more complicated. It is known that the multicast routing problem in traditional IP multicast networks reduces to the NP-hard Steiner tree problem [12]. However, when network coding is used in multicast routing, this problem becomes much simpler. In such a case, an immediate node along with a corresponding capacity allocation scheme. We will also analyze the statistical characteristics of encoding nodes and maximum flow in ad-hoc networks based on random graph theory.

The theory and application of network coding have been studied by many researchers. Koetter and Médard [2] proposed an algebraic approach to network coding and reduced it to solving a set of linear equations. Jaggi and Sanders developed a polynomial time algorithm for network code construction in [7]. Unfortunately, a deterministic coding algorithm is generally accompanied by high complexity. Moreover, it requires complete information about the whole network topology and consequently, is difficult to implement in a distributed mode. However, if a negligible probability of decoding failure at the receivers is permitted, the data flows can be encoded randomly and distributedly. Ho and Médard ([4],[5]) investigated the randomized coding approach for multicast. Furthermore, Chou and Wu proposed a practical scheme on network coding by applying random coding theory [8].

In some networks that can be characterized by directed graphs, it has been shown that network coding can achieve the maximum multicast capacity, which is bounded by the minimum cut between the source and each destination [3]. Actually, the maximum flow routes from a source to a destination can be found efficiently in directed graphs by some well developed algorithms, such as the Ford-Fulkerson algorithm and the Dinic algorithm [12]. Based on the Ford-Fulkerson algorithm, we will put forward a practical distributed scheme for finding the maximum flow multicast routes in wireless ad-hoc networks. For undirected graphs, we shall propose a modified Ford-Fulkerson algorithm on maximum flow routing, where we assume that each undirected edge has unit capacity. The algorithm can also be used in a network with integral edge capacities. But for undirected graphs, there exists the Conflict Phenomenon in multicast routing, which is a holdback of achieving the maximum multicast capacity. An alternative method using the minimum cost flow algorithm was given by [13] to alleviate the effect of this phenomenon. In this paper, we will show that under certain conditions, the multicast routing problem in undirected networks is NP-hard.

For completely random network coding, every relay between the source and destinations encode its input data flows, which is very uneconomical. To reduce the cost and complexity of network coding, one can identify the nodes that need encoding, and only perform the coding operations at these nodes, rather than throughout the whole network. In [6], Fragouli et al gave a definition of encoding nodes by subtree decomposition. In [13], a concrete approach for searching...
the encoding nodes in a network with unit edge capacity
was given. In this paper, we will consider the network with
total edge capacity and present the sufficient and necessary
conditions under which a node becomes an encoding node.
Furthermore, we will consider the capacity allocation problem
in order to minimize the value of local encoding flow at a node.

Computing the minimum number of encoding edges in a
multicast network has been shown to be an NP-hard problem
by Langberg et al in [9]. In this paper, we will further discuss
the statistical properties of the number of encoding nodes and
edges. A random graph model will be used to characterize
wireless ad-hoc networks. Specifically, some terminals will
be assumed to be randomly located in a given region, and
each one of them has direct links with the terminals inside
its transmission area [10]. We will show that as the network
size becomes very large, the probability that a node on the
routes needs encoding becomes significantly small. Moreover,
the ratio of the number of encoding nodes and the destination
number \( m \) can be bounded by \( O(m^2) \). The conclusion here
can be used to estimate the cost of network coding in a random
network.

This paper is organized as follows. In Section II, we
will introduce the random graph model for wireless ad-hoc
networks and some notations used in this paper. In Section
III, we present a practical distributed maximum flow routing
algorithm for ad-hoc networks. We will also prove that the
maximum flow multicast routing problem in undirected graphs
is NP-Hard under certain conditions. In Section IV, the con-
tion of a node to be an encoding node and the capacity
allocation scheme will be presented. In Section V we will
make a theoretical analysis of the statistical properties of
the number of encoding nodes and edges. Section VI presents
the sample simulation results. Finally, we conclude the paper in
Section VII.

II. WIRELESS AD-HOC MULTICAST NETWORK MODEL

In a wireless ad-hoc network, the terminals are randomly
located in a certain area, and communicate to each other ac-
cording to some transmission protocol. Since the transmission
power of a terminal is limited, each terminal has a covering
radius, which is determined by the inverse power law model
of attenuation and a predetermined threshold of power level
for successful reception. That is,

\[
P_0 \left( \frac{R}{d_0} \right)^{-\alpha} = P_{\text{threshold}}
\]

where \( P_0 \) is the power received at a close-in reference point
in the far field region of the antenna at a distance \( d_0 \) from
the transmitting antenna and \( \alpha \) is the path loss. Assuming a
symmetrical scenario, then all the terminals will have the same
transmission power and thus the same covering radius \( R \). Two
terminals have a direct link between them if their distance
is smaller than \( R \). For each terminal, the nodes that have
direct links to it are called its neighbors. We adopt a random
graph \( G(\mathcal{V}, \mathcal{E}) \) to model a wireless ad-hoc network. The node
set \( \mathcal{V} = \{v_1, v_2, \ldots, v_n\} \) shall denote the \( n \) terminals, which
are assumed to be independently and uniformly distributed
in an \( 1 \times 1 \) square region. There is an edge between node
\( v_i \) and \( v_j \), if their distance \( D(i, j) \) is smaller than \( R \). Thus,
the edge set \( \mathcal{E} = \{(v_i, v_j) | D(i, j) \leq R, 0 < i, j < n \} \).

For convenience, we introduce some further definitions. The
number of the links connected to node \( v_i \) is defined as its
degree, denoted by \( d(i) \). The transmission capacity and the
real data rate of edge \( (v_i, v_j) \) is denoted by \( C(i, j) \) and \( f(i, j) \),
respectively, satisfying \( 0 \leq f(i, j) \leq C(i, j) \).

In a multicast session, the source node \( s \) transmits multiple data packets to a set of destinations \( T = \{t_1, \ldots, t_m\} \).
\( F_s \) is the maximum flow value from the source \( s \) to the destination node \( t_i \) and the multicast capacity \( F \) represents the maximum data rate of the multicast session.

In general, \( C(i, j) \) can be approximated by a positive integer
through proper scaling. If the edge \( (v_i, v_j) \) does not exist,
we let \( C(i, j) = 0 \). Based on the above description, an edge
\( (v_i, v_j) \) can be divided into \( C(i, j) \) edges of unit capacity. By
adding a node to each of these edges, a new graph \( G^* \) with
edges of unit capacity can be formed. The multicast capacity
\( F \) and all \( F_i \) in \( G^* \) are the same as those in \( G \) [12]. A graph
of unity edge capacity is defined as a Simple Graph. By graph
theory, we know that a common graph, whose edge capacities
are integers less than a constant \( c \), can be transformed into a
simple graph in polynomial time. It turns out that, the flow
analysis problem in a simple graph is more complicated than
that in a common graph. If the flow analysis problem in simple
graphs can be solved, it can also be solved in common graphs.

In the following discussion, the graph model is assumed to be
a simple graph, if there is no special statements.

The well known maximum-flow minimum-cut theorem states
that the maximum flow \( F_i \) from \( s \) to \( t_i \) is equal to the
value of the minimum cut between them. That is,

\[
F_i = \min \{ C(\mathcal{V}', \mathcal{V} \backslash \mathcal{V}') | s \in \mathcal{V}', t_i \in \mathcal{V} \backslash \mathcal{V}' \}
\]

In an underlying directed topology graph, \( F \) is equal to
the minimum value of the minimum cuts of all the source-
destination pairs. That is,

\[
F = \min \{ \min \{ C(\mathcal{V}', \mathcal{V} \backslash \mathcal{V}') \} \}
\]

It has been proved that to achieve this limit, linear network
coding is sufficient [3]. But in a multicast network with
an underlying undirected graph, such as half-duplex ad-hoc
network, this limit may not be achieved by any coding method
due to the Conflict Phenomenon [13]. It will be shown that
if the destinations can only receive data, the calculation of
the multicast capacity in an undirected graph is an NP-hard
problem.

III. MULTICAST ROUTING IN AD-HOC NETWORKS BASED ON NETWORK CODING

From Eqn. (2), we can conclude that by employing network
coding in wireless ad-hoc networks, the multicast routing
problem can be divided into \( m \) searching operations for
the maximum flow from source \( s \) to destination \( \{t_j\}_{j=1}^{m} \).

According to the transmission mode, the network can be
modeled by directed and undirected graphs, corresponding to
full and half-duplex modes, respectively. In this part, we will
discuss the multicast routing problem in both directed and
undirected graphs. A practical approach for ad-hoc networks
will also be presented.
A. Maximum Flow Routing

The Ford-Fulkerson Algorithm was proposed in 1956 to solve the maximum flow problem. Since then, many efficient algorithms have been developed in directed graphs. To compute the maximum flow in an undirected graph, one can transform the undirected graph into a directed one by simply replacing each undirected edge by two inverse directed arcs with the same capacity. The maximum flow in the obtained graph is then equal to that of the original undirected graph [15]. In the following, a maximum flow routing algorithm in a simple undirected graph based on the Ford-Fulkerson algorithm is given in [13]. It is easy to see that as the algorithm terminates, \( F \) is different edge disjoint paths from \( s \) to \( t \) are obtained.

**Step 0:** Initialization.

Set the flow to be 0 on all edges. Then, mark \( s \) and set \( s \) to be unchecked.

**Step 1:** Check each incident edge \((v_i, v_k)\) of a marked vertex \(v_i\). If \( v_k \) has not been marked, then

1. If \( f(i, k) = 0 \) and \( f(k, i) = 0 \), mark \( v_k \) by \( i \);
2. If \( f(k, i) = 1 \), mark \( v_k \) by \(-i\);

After all the edges \((v_i, v_k)\) have been checked, set \( i \) to be unchecked.

**Step 2:** Checking the destination.

1. If the destination node \( t \) has been marked, it indicates that an augmenting path has been found.

Then, the flow can be increased by 1 along this path. First, set \( v_j = t \), \( i = The Mark of t \).

1.1) If the mark of \( v_j \) is positive, then increase \( f(i, j) \) by 1; If the mark of \( v_j \) is negative, then decrease \( f(j, i) \) by 1.

1.2) Set \( v_j = v_i \), \( i = The Mark of j \).

1.3) If \( v_j = s \), mark \( s \), and set other nodes to be unmarked, go to step 1; Otherwise, go to (1.1).

2) If \( t \) has not been marked, and there exists an unchecked node, go to step 1.

3) Otherwise, stop.

If the edge capacities of the undirected graph are not equal to 1, one can adjust the algorithm to meet this condition. Let \( \delta \) denote the flow value of the augmenting path at node \( v_i \). At the beginning, set \( \delta_s = \infty \). (1) and (3) in Step 1 can be modified as follows.

1) If \( f(i, k) < C(i, k) \) and \( f(k, i) = 0 \), mark \( v_k \) by \( i \), and set \( \delta_k = min(\delta_k, C(i, k) - f(k, i)) \);
2) If \( f(k, i) > 0 \), mark \( k \) with \(-i\), and set \( \delta_k = min(\delta_k, f(k, i)) \);

In Step 2, if \( t \) has been marked, the flow can be increased by \( \delta_t \) along the augmenting path. Clearly, the proposed algorithm as described above can avoid loops on the maximum flow routes.

B. A Practical Routing Approach for Wireless Ad-Hoc Networks

Generally speaking, there are two types of maximum flow algorithms. One is the Ford-Fulkerson type, which maintains the conservation of flow at each node except for the source and the destination. It means that at each intermediate node, the input and output flow values are equal to each other. In this case, the flow value is gradually increased until it reaches the upper bound. The other one is the preflow-push method. In contrast to the former one, algorithms of this type do not guarantee the conservation of flow. The value of the flow on each edge is amended iteratively. When all the intermediate nodes satisfy the demand of flow conservation, the maximum flow is found. Strictly speaking, both type of algorithms can be implemented in a distributed fashion. But it is easier to control the flow value by the Ford-Fulkerson type algorithms. Therefore, our proposed practical approach will be based on the Ford-Fulkerson algorithm.

Throughout this section, we shall assume that the channel between each pair of terminals in a wireless ad-hoc network is symmetric and half duplex.\(^1\)

While computing the maximum flow routes from \( s \) to \( t \), each intermediate node must stay at one of the following three states: 0-Waiting, 1-Receiving, and 2-Forwarding. The main idea of this routing algorithm is to search the augmenting paths iteratively in the network by exchanging the Request Message (RM) packet among the nodes. The RM packet contains the destination node’s ID \( j \) and the current flow value \( \varepsilon \). The process of searching out one augmenting path shall be called a Phase. At the beginning of each phase, the source \( s \) broadcasts an RM to all of its neighbors. Each intermediate node \( v_i \) records the value of the current flow to destination \( t_j \) as \( \varepsilon_i^r \) and sets itself to state 0. To know whether an RM packet is outdated, each \( v_i \) needs a flag \( \phi_i^r \). Initially let \( \phi_i^r = 1 \), \( \varepsilon_i^r = 0 \), \( \phi_i^s = 0 \), and flow \( f(i, k) = 0 \) for all \((v_i, v_k)\).

In the proposed routing algorithm, each intermediate node \( v_i \) will perform the following operations in a distributed mode.

**State-0:** if an RM packet from node \( v_k \) arrives, go to State-1.

**State-1:** Receive the RM packet with destination ID \( j \). Then check \( \varepsilon \) and \( \phi_s^r \):

1) If \( \varepsilon = \varepsilon_i^r \) and \( \phi_i^r \leq \phi_i^s \), discard the packet and go to State-0;
2) If \( 0 < \varepsilon < \varepsilon_i^r \), discard the packet and go to State-0;
3) Set \( \varepsilon_i^r = \varepsilon \) and \( \phi_i^r = \phi_i^s - 1 \)
   - If \( f(k, i) = f(i, k) = 0 \), mark \( v_i \) by \( k \);
   - If \( f(k, i) = 1 \), mark \( v_i \) by \(-k\);
   - Otherwise, discard the packet and go to State-0.
4) Let \( \phi_i^s = \phi_i^r \).
   - Go to State-2.

**State-2:** Forward the RM packet to all its neighbors. Then go to State-0.

Once \( t_j \) receives an RM packet that is not outdated, an augmenting path is found. \( t_j \) will then send an Acknowledgement (ACK) packet to \( s \). Upon receiving the ACK successfully, \( s \) will update \( \varepsilon \), \( \phi_i^s \) as well as the routing table. Then it will send out another RM to continue searching for a new augmenting path until the maximum flow routes are obtained. Since the channel is assumed to be symmetric, the ACK packet can be transmitted back along the augmenting path and the flow

---

\(^1\)For the full-duplex networks, we only need to replace each undirected edge by a pair of directed edges with reverse directions.
values on the links along the path are updated according to the mark of each intermediate node. Specifically, if the mark of \( v_i \) is \( k \), let \( f(k, i) = 1 \). Otherwise, if the mark of \( v_i \) is \(-k\), let \( f(i, k) = 0 \). The ACK is then passed to \( v_k \). If the channels are not symmetrical, \( t_j \) must find another route to \( s \) for ACK, and the flow values on the augmenting path must be updated after \( s \) has received the ACK packet.

In the approach above, \( \phi^e \) is utilized to distinguish the new searching operation from the old one. Since in wireless ad-hoc networks, the network topology may change from time to time due to the breakdown or movement of the terminals. If \( s \) detects a change that has effect on the maximum flow routes, or the search fails in some phase, \( s \) will increase \( \phi^e \) by 1, and start a new searching phase with \( \varepsilon \) unchanged. Because \( \phi^e > \phi^e \) for all \( i \) now, all the intermediate nodes know that it is a new RM and will not discard it casually.

It can be easily proved that the proposed routing algorithm terminates if and only if the maximum flow routes are found. Otherwise, if there exist some augmenting paths along which an RM can reach \( t_j \), the flow can be further increased. In fact, when the maximum flow value is obtained, it can not find any augmenting path unless the topology of the network changes. It should be also noted that each time \( s \) sends out an RM to search for an augmenting path, the packet will arrive at \( t_j \) along the augmenting path with the shortest delay. In other words, the proposed algorithm has the property of the shortest path first. That is to say, each augmenting path by our proposed method must be the time-shortest augmenting path in the network with the current flow. Finally, if we assume that the antenna is omnidirectional, then when an intermediate node forwards the RM packet, broadcast is sufficient. Besides, every intermediate node needs to forward the RM for at most one time in each phase.

One solution for multicast routing is to run the above algorithm for \(|T|\) times and select \( F^* \leq \min \{ F_t \} \) as the multicast data rate. But this scheme will take too much time and the RM packets may occupy a large amount of bandwidth. It is better to search out the maximum flow multicast routes for all the destinations simultaneously. Thus, in each phase, we must find an augmenting path for each destination, and every intermediate node shall need to record the flow value on all of its incident links for each \( t_j \in T \). To mark the augmenting path for different destinations, we add the IDs of all \( t_j \in T \) to the RM packet sent by \( s \). Upon receiving an RM through the link \((v_k, v_i)\), the intermediate node \( v_i \) will delete the IDs of the destinations, if the flow of them cannot be increased on \((v_k, v_i)\). Then it will forward the RM with the rest IDs. Here the initial state of the network is the same as that of the above algorithm, except that \( f^l(i, k) = 0 \) for all the links \((v_i, v_k)\) and \( t_j \in T \). In one phase, each intermediate node \( v_i \) (including the destinations, for they may act as relays) will run the algorithm as follows.

**State-0:** if a RM packet from node \( v_k \) arrives, go to State-1.

**State-1:** Receive the RM packet. Then check \( \varepsilon \) and \( \phi^e \):

1. (1.1) If \( \varepsilon < \varepsilon_i \), discard the packet and go to State-0;
   
2. (1.2) If \( \varepsilon = \varepsilon_i \) and \( \phi^e = \phi^e_i - 1 \), discard the packet and go to State-0;

3. (1.3) Set \( \varepsilon_i = \varepsilon \) and \( \phi^e_i = \phi^e_i - 1 \). For each ID \( j \) in the RM
   
   - If \( j \) already exists in \( v_i \)’s routing table, remove \( t_j \)’s ID from the RM;
   
   - If \( f^l(i, k) = 1 \), remove \( t_j \)’s ID from the RM;

   - If \( f^l(i, k) = f^l(i, k) = 0 \), add \( j \& k \) to \( v_i \)’s routing table;

   - If \( f^l(i, k) = 1 \), add \( j \& k \) to \( v_i \)’s routing table;

   - If there is no ID in the RM, discard the packet and go to State-0.

   Go to State-2.

**State-2:** If \( v_i \) is the destination \( t_i \), remove \( l \) from the RM, and send ACK back to \( s \).

- If there is no ID in the RM, discard the packet and go to State-0.

   Forward the RM packet to all its neighbors. Then go to State-0.

As in the case of the former algorithm, the multicast flow value is also gradually increased here. Thus, \( s \) can control the flow value. In practice, some destinations’ ACKs may not reach \( s \) in time due to delay or link failures. In this case, \( s \) can send another RM to search the augmenting paths for these destinations until all the ACKs reach \( s \).

Note that in full-duplex networks, which is modeled by a directed graph, this algorithm can search out the maximum flow multicast routes which is specified by Formula (2). But in a half-duplex network, sometimes the maximum flow multicast routes cannot be obtained due to the Conflict Phenomenon [13]. We will further discuss this problem in the next subsection.

**C. Multicast Routing In Undirected Graphs And Conflict Phenomenon**

Fig. 1 [13] indicates that in undirected graphs, the maximum flow value defined in Eqn. (2) may not be achieved due to the Conflict Phenomenon. It is not hard to check that the maximum flow values from \( s \) to \( t_1 \) and \( t_2 \) are both equal to 2, but the routing paths for each of them pass through one common edge \( AB \) from two completely reverse directions. Thus, the realized multicast capacity is less than
2. It is therefore easy to see that computing \( F \) in wireless ad-hoc multicast networks is not so easy. To the best of our knowledge, this is still an open problem. In this part, we consider a special case, where the destinations are not allowed to forward messages to other nodes. The following theorem shows that under this condition, computing \( F \) in an undirected graph or wireless half-duplex ad-hoc network is an NP-hard problem.

**Theorem 1:** If all the edges linked to destinations \( \{ t_i : 0 < i \leq m \} \) are restricted to be the incoming edges, and no direct links between any two destinations exist, then computing the maximum multicast flow \( F \) in such an undirected graph is an NP-hard problem.

The proof of Theorem 1 is given in Appendix A.

**IV. ENCODING NODES AND CAPACITY ALLOCATION**

In a multicast session, the maximum flow routes from \( s \) to different destinations, may pass through common edges. This means that the flows for different \( t_j \) may share some common links. How to simplify queuing and scheduling such that each node can maintain the same transmission data rate as that of its input flows? To solve this problem, network coding will be used. In wireless networks, the relay nodes can either decode all the input data into information bits and encode them together into one flow, or combine the physical signals of different input data by so-called Physical-Layer Network Coding [14]. The latter method needs no time or frequency division. In particular, to perform network coding at all of these nodes will bring about high cost and complexity. In this part, we will present the condition under which a node must encode the flows for different destinations before forwarding them as one flow with the same transmission rate.

The following lemma presents a sufficient and necessary condition for a node to be an *Encoding Node* in a simple graph [13].

**Lemma 1:** A node is an encoding node if and only if there exist more than one incoming flows for different destinations carried by different incoming edges and sharing a common outgoing edge.

For a common graph, consider node \( A \) as shown in Fig. 2, where there are \( L \) flows passing through different incoming edges of \( A \) for destination sets \( \{ T_j \}_{j=1}^{L} \) with all the flows sharing a common outgoing edge. Let \( C \) denote the outgoing edge’s capacity, and \( f_1, f_2, \ldots, f_L \) denote the corresponding values of the \( L \) flows. For convenience, \( f_1, f_2, \ldots, f_L \) and \( C \) are assumed to be integers. According to Lemma 1, the following theorem holds:

**Theorem 2:** A node \( A \) is an encoding node if and only if

\[
\sum_{j=1}^{L} f_j > C.
\]

Note that even if node \( A \) is an encoding node, it may not be necessary to encode all the flows \( f_1, f_2, \ldots, f_L \) in most cases. As shown in Fig. 3, two incoming flows will share a common outgoing edge of capacity \( C = 12 \). In this case, \( f_1 = 8, f_2 = 6 \), and hence only \( f_1 + f_2 - C = 2 \) of each flow shall need to be encoded. Thus, a proper capacity allocation scheme is required to minimize the quantity of flow to be encoded, and consequently to reduce the complexity of network coding.

Without loss of generality, assume that \( C \geq f_1 \geq f_2 \geq \ldots \geq f_L \). On the outgoing edge, the link capacity is divided into \( C \) unit parts. The encoded part of all the flows will occupy \( g \geq 0 \) units, and the rest \( C - g \) parts will still carry the original incoming flow without encoding. To compute \( \min(g) \), we first introduce the following lemma.

**Lemma 2:** Let \( i \) be a positive integer such that \( i \leq L \) and \( g_i \) denote the minimum value of encoded flow given only \( i \) incoming flows, namely, \( f_1, f_2, \ldots, f_i \). Then, we have \( g_i \leq g_{i+1} \).

**Proof:**

Suppose that for some \( i, g_i > g_{i+1} \). If removing flow \( f_{i+1} \) from the flow set \( \{ f_1, f_2, \ldots, f_i \} \), the resulted value of encoded flow will not increase. This implies that, the resulted value of encoded flow satisfies \( g' = g_{i+1} < g_i \). On the other hand, since \( g_i \) is assumed to be the minimum value of encoded flow given \( f_1, f_2, \ldots, f_i, g' \geq g_i \). This results in a contradiction. □

Based on Lemma 2, we have the following theorem on the flow allocation for encoding.

**Theorem 3:** Suppose that \( C \geq f_1 \geq f_2 \geq \ldots \geq f_L \geq f_{L+1} = 0 \), and \( T_i \cap T_j = \phi \) for \( 0 < i, j \leq L \) and \( i \neq j \). If there exists an integer \( 1 < l \leq L \), such that \( \sum_{i=1}^{l-1} f_i > C \) and \( \sum_{i=1}^{l} f_i \leq C \), then the minimum value of encoded flow on the outgoing edge is

\[
g = \max\{ \sum_{i=1}^{l} f_i - C, f_{l+1} \}.
\]
Proof:

By Lemma 2, \( g = g_L \geq g_{L-1} \geq \ldots \geq g_1 = 0 \). Since \( \sum_{i=1}^{L} f_i \leq C \), then by Theorem 2, \( g_1 = g_2 = \ldots = g_{l-1} = 0 \).

On the other hand, \( \sum_{i=1}^{L} f_i > C \). This implies that \( \sum_{i=1}^{L} f_i - C \) of flow \( f_i \) must be encoded with other flows. Thus, \( g \geq g_l = \sum_{i=1}^{L} f_i - C \). Let us consider the flow \( f_{l+1} \) in the following two cases.

Case 1: \( f_{l+1} \leq \sum_{i=1}^{l} f_i - C \)

In this case, flow \( f_{l+1} \) can be encoded together with the flows of value \( \sum_{i=1}^{l} f_i - C \) that have already been encoded. Next since \( f_i \leq f_{l+1} \), for \( i \geq l+1 \), then the minimum encoded flow value on the outgoing edge is \( g = \sum_{i=1}^{l} f_i - C \).

Case 2: \( f_{l+1} > \sum_{i=1}^{l} f_i - C \)

In this case, the \( \sum_{i=1}^{l} f_i - C \) of flow \( f_{l+1} \), which can be encoded with the flows that are already encoded, the remaining \( f_{l+1} - (\sum_{i=1}^{l} f_i - C) \) of flow \( f_{l+1} \) must be encoded with the flows which were not encoded previously. Thus, we have \( g_{l+1} = f_{l+1} \). Since \( f_i \leq f_{l+1} \), for \( i \geq l+1 \), then the minimum value of encoded flow on the outgoing edge is given by \( g = f_{l+1} \).

By combining these two cases, it follows that \( g = \max \{ \sum_{i=1}^{l} f_i - C, f_{l+1} \} \).

In order to reduce the encoded amount of each flow \( f_i \), a greedy algorithm on the capacity allocation can be developed according to the proof of Theorem 3. The detailed description of this algorithm is presented in Appendix B.

V. STATISTICAL ANALYSIS ON THE ENCODING NODE AND EDGE NUMBER

In a simple random graph, the edges are assumed to be independently and identically distributed with probability \( p \). If there are only two destinations \( t_1 \) and \( t_2 \), the probability that node \( v \) needs encoding is given by

\[
\Pr_{en} = \sum_{k=1}^{n-1} p(d(v)=k) \sum_{i=0}^{\lfloor k/2 \rfloor} p_{r_1}(i)d(v)=k \sum_{j=0}^{\lfloor k/2 \rfloor} p_{r_2}(j)d(v)=k)\rho_c
\]

where \( \lfloor x \rfloor \) denotes the least integer greater or equal to \( x \), \( p_c \) is the probability that node \( v \) needs encoding given that the degree of \( v \) is \( d(v)=k \), and the flow for \( t_1 \) and \( t_2 \) occupies \( 2i \) and \( 2j \) edges of node \( v \), respectively. That is, \( i (i \geq 0) \) incoming edges and \( j \) outgoing edges for the flow of \( t_1 \), while \( j \) incoming and \( j \) outgoing edges for \( t_2 \). The probability that the multicast flow for one destination occupies \( 2i \) out of node \( v \)'s \( k \) incident edges is given by [13]

\[
P_t(i|d(v)=k) = C_k^i p_c^{2i}(1-p_c)^{k-2i}
\]

where \( p_c \) is the probability that the maximum flow routes pass through one edge. In a completely random graph, \( p_c \) is identical for all the edges. The following theorem gives an upper bound on \( p_c \).

Theorem 4: In a random graph \( G = (V, E) \), \( |V| = n \). If \( (n-1)p > 1 \) (i.e., \( n \) is large enough), then for any \( \epsilon > \frac{1}{\sqrt{(n-1)p}-1} \) and \( \epsilon < 32/33 \), the probability that the maximum flow routes pass through an edge satisfies \( p_c < \frac{n/\sqrt{(n-1)p}-1}{(1-\epsilon)n/2\sqrt{(n-1)p}} \).

Proof:

Let \( F_t \) be the maximum flow from the source \( s \) to a destination \( t \). Consider the subgraph \( \hat{G} \) which consists of \( s \), \( t \), and all the nodes on the maximum flow routes along with the corresponding edges on the \( F_t \) edge disjoint paths. Then \( \hat{G} = (\mathcal{V}, \mathcal{E}) \subseteq G \). According to [17], the length of the shortest path from \( s \) to \( t \) in \( \hat{G} \), which corresponds to the edge number of the shortest path, satisfies

\[
L_1 = D(s, t) \leq \frac{2|\mathcal{V}|}{\sqrt{|F_t|}}
\]

By removing the shortest path in \( \hat{G} \), the maximum flow becomes \( F_{t-1} \). Consequently, the length of the shortest path in the remaining subgraph satisfies \( L_2 \leq \frac{2|\mathcal{V}|}{\sqrt{|F_{t-1}|}} \). By repeating this operation, we can determine the total length of the maximum flow routes in \( G \) as

\[
L \leq \frac{2|\mathcal{V}|}{\sqrt{|F_t|}} + \frac{2|\mathcal{V}|}{\sqrt{|F_{t-1}|}} + \ldots + \frac{2|\mathcal{V}|}{\sqrt{|F_1|}} + \text{the rest four paths’ lengths.}
\]

Note that each path length is no more than \(|\mathcal{V}|\) and \(|\hat{\mathcal{V}}| < |\mathcal{V}|\). Thus,

\[
L \leq 2|\mathcal{V}|((\frac{1}{\sqrt{|F_t|}} + \frac{1}{\sqrt{|F_{t-1}|}} + \ldots + \frac{1}{\sqrt{5}}) + 4|\mathcal{V}|
\]

\[
< 2|\mathcal{V}| \int_4^{F_t} \frac{1}{\sqrt{f}} df + 4|\mathcal{V}|
\]

\[
= 4|\mathcal{V}|(\sqrt{F_t} - 2) + 4|\mathcal{V}|
\]

\[
= 4n(\sqrt{F_t} - 1).
\]

Since \( E\{F_t\} \leq (n-1)p \) [13], it follows that

\[
E\{\sqrt{F_t} - \sqrt{(n-1)p}\} = E\{\sqrt{F_t} - (n-1)p\} \sqrt{F_t + \sqrt{(n-1)p}} \leq 0.
\]

As a result,

\[
E\{L\} \leq 4n(\sqrt{(n-1)p} - 1).
\]

Let \( S \) denote the edge number in \( G \), then \( E(S) = (n-1)p/2 \) and \( D(S) = n(n-1)p(1-p)/2 \). By the Chebyshev inequality, we have

\[
P(S < (1-\epsilon)E(S)) \leq \frac{D(S)}{2(1-\epsilon)^2} = \left(\frac{1 - \epsilon}{\epsilon^2 n(n-1)p}\right)^2.
\]
Thus,

\[ p_e = E\left(\frac{L}{S}\right) \]
\[ = E\left(\frac{L}{S} | S \geq (1 - \epsilon)E(S)\right) P(S \geq (1 - \epsilon)E(S)) \]
\[ + E\left(\frac{L}{S} | S < (1 - \epsilon)E(S)\right) P(S < (1 - \epsilon)E(S)) \]
\[ < \frac{E\{L\}}{(1 - \epsilon)E\{S\}} + P(S < (1 - \epsilon)E(S)) \]
\[ < \frac{8n(\sqrt{n(n-1)p-1})}{(1 - \epsilon)n(n-1)p} + \frac{1 - p}{\epsilon^2 n(n-1)p} \]
\[ = \frac{8(\sqrt{n(n-1)p-1})}{n(n-1)p} \left(1 - \epsilon + \frac{1 - p}{8\epsilon^2 n(\sqrt{n(n-1)p-1})}\right). \]

Since \( \epsilon \geq \frac{1}{\sqrt{n(n-1)p-1}} \) and \( np > (\sqrt{n(n-1)p-1})^2 \), we have

\[ \frac{1 - p}{8\epsilon^2 n(\sqrt{n(n-1)p-1})} < \frac{(1 - p)p}{8(\sqrt{n(n-1)p-1})} \]
\[ < \frac{(1 - p)p}{8} \]
\[ \leq \epsilon/32 \]
\[ < \frac{\epsilon}{32(1 - \epsilon)(1 - 33/32\epsilon)}. \]

Thus,

\[ p_e < \frac{8(\sqrt{n(n-1)p-1})}{n(n-1)p} \left(1 + \frac{\epsilon}{32(1 - \epsilon)(1 - 33/32\epsilon)}\right) \]
\[ = \frac{8(\sqrt{n(n-1)p-1})}{(1 - 33/32\epsilon)(n-1)p}. \]

The result in Theorem 4 implicitly indicates that if there are a large number of nodes in a multicast network, the probability that a node is an encoding node is very small. Therefore, performing network coding only at the encoding nodes can significantly reduce the implementation complexity.

Next, we consider the probability that an edge is an encoding edge, and obtain the following theorem on \( p_{ed} \).

**Theorem 5:** Assume that \( |T| = m \), \( m \ll n \) the probability that an edge is an encoding edge satisfies \( p_{ed} < \frac{m(m-1)p^2}{2} - O(p^3) \).

**Proof:**

For a random graph and a random multicast session, the occupancy of an edge by the maximum flow route of each destination can be seen to be independent and equiprobable. Thus, given an edge \((v_i, v_j)\), suppose the multicast routes of \( k \) destinations whose pass \((v_i, v_j)\). Then \( k \) approximately obeys the binomial distribution. If \( k = 0 \) or \( k = 1 \), the edge \((v_i, v_j)\) need not be an encoding edge. Thus,

\[ p_{ed} \leq 1 - p(k = 0) - p(k = 1) \]
\[ = 1 - (1 - p_e)^m - mp_e(1 - p_e)^{m-1} \]
\[ = 1 - (1 - p_e)^m - \frac{(m-1)(m-2)}{2} p_e^2 + \ldots \]
\[ \times [1 + (m-1)p_e] \]
\[ = 1 - [1 + (m-1)p_e - (m-1)p_e - (m-1)^2 p_e^2 \]
\[ + \frac{(m-1)(m-2)}{2} p_e^2 + O(p^3)] \]
\[ = \frac{m(m-1)}{2} p_e^2 - O(p^3). \]

This theorem indicates the probability that an edge is an encoding edge can be bounded by the square of the number of destinations. Since the encoding node number is no more than the number of encoding edges, \( p_{en} \) also has this upper bound.

**VI. SIMULATION RESULTS**

To set up a random graph model, We first randomly create some vertices that are located uniformly and independently in \([0, 1] \times [0, 1]\). The covering radius of the nodes is set to be \( R \). Here we first assume \( n = 100 \), and the number of destinations, denoted by \( t_1 \) and \( t_2 \), to be 2. In this section, we will implement the routing algorithm mentioned in Section III in random graphs, and investigate the performance of the algorithm as well as the statistical property on encoding edge number by simulation.

We shall consider two different cases of the distribution of transmission delay of each link. In the first one, the transmission delay of each link is assumed to be independently and uniformly distributed. But the delay of all the links is fixed during one multicast session. In the second case, the transmission delay of each link is a constant. To compare the effect of the transmission delay on the system performance, it assumes that the average values of the transmission delay of
all the links to be the same in both cases. Fig. 4 presents the delay for finding the routes of unit flow using the practical algorithm introduced in Section III-B. Here the dashed line denotes the case where the delay of each link is independently and uniformly distributed from 5 to 15 time units. Likewise, the solid line denotes the case where the delay of each link is equal to 10. The results listed in Fig. 5 demonstrate that the delay of the former one is smaller than that of the latter, which validates the shortest path first property of our proposed routing selection algorithm. Fig. 4 also indicates that the average delay time decreases as the covering radius $R$ increases. This is because the bigger the covering radius $R$ is, the larger the number of edges in the wireless ad-hoc network is. Thus, resulting in a smaller average path length.

Fig. 5 shows the distribution of the encoding edge number with different destination numbers. The solid line marked by squares is the curve for the network with $n_1 = 200$, and the dashed curve marked by stars is for the network with $n_2 = 100$. It can be seen that as the number of destination nodes increases, the number of encoding edges also increases. Moreover, the encoding edge number in the network with $n_1 = 200$ is approximately twice that of with $n_2 = 100$. Suppose the edge number in these two networks obeys binomial distribution $B((n_1 - 1)n_1/2, p_1)$ and $B((n_2 - 1)n_2/2, p_2)$, respectively, then the edge number $S_1$ in the former network is about $4p_1/p_2$ times $S_2$ of the latter network. From Section V, the probability that an edge is an encoding edge approximately satisfies $p_{ed} \propto p_2^2 \propto (\frac{1}{(n-1)p})^2$. Thus, $p_{ed}$ in the former network is about $p_2/(2p_1)$ of that in the latter one. And the average number of encoding edges is equal to $S_{p_{ed}}$. This is consistent with the simulation result. Fig. 6 shows the probability of conflict phenomenon versus different destination number $|T|$. $p_c$ increases as the destination number increases, according with our intuition. But the network size and complexity have little influence on this phenomenon.

Table I shows the simulation data for both $t_1$ and $t_2$ under different values of $R$. The numbers in the last column denote the frequencies of Conflict Phenomenon. From the table, it can be seen that when $R$ is not too small ($> 0.1$), the probability of path confliction decreases as $R$ increases. The average probability of path confliction is around 10%. Hence, we can see that the conflict phenomenon has a significant effect on the multicast capacity in undirected graphs, and thus in wireless half-duplex ad-hoc multicast networks.

<table>
<thead>
<tr>
<th>$R$</th>
<th>Average maximum flow of $t_1$</th>
<th>Average maximum flow of $t_2$</th>
<th>Average maximum flow of multicast</th>
<th>Average encoding node number</th>
<th>Conflict (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.15</td>
<td>0.292</td>
<td>0.237</td>
<td>0.038</td>
<td>0</td>
<td>2.7</td>
</tr>
<tr>
<td>0.25</td>
<td>3.268</td>
<td>3.197</td>
<td>1.893</td>
<td>0.4481</td>
<td>14.2</td>
</tr>
<tr>
<td>0.3</td>
<td>8.134</td>
<td>8.118</td>
<td>6.784</td>
<td>1.6313</td>
<td>14.3</td>
</tr>
<tr>
<td>0.4</td>
<td>12.436</td>
<td>12.583</td>
<td>10.792</td>
<td>1.6022</td>
<td>10.5</td>
</tr>
<tr>
<td>0.5</td>
<td>17.203</td>
<td>17.26</td>
<td>15.027</td>
<td>1.2415</td>
<td>9.3</td>
</tr>
<tr>
<td>0.35</td>
<td>22.485</td>
<td>22.516</td>
<td>19.7406</td>
<td>0.912</td>
<td>7.1</td>
</tr>
</tbody>
</table>

VII. CONCLUSION

In this paper, we proposed a practical routing scheme with the shortest path first property. We also investigated the Conflict Phenomenon which happens in multicast routing in undirected graphs while applying network coding. We proved that computing the maximum multicast flow in an undirected graph is NP-hard if the destinations are only allowed to receive messages. In order to reduce the value of encoded information flows and hence the network coding complexity, we considered the capacity allocation problem, and presented a greedy algorithm to solve it.

Throughout this paper, we adopted a random graph to approximately model a wireless ad-hoc network. Based on this model, some theoretical analysis and simulation results on the distribution of the number of encoding nodes and edges have been presented. It was shown that as the network size becomes larger, the probability that a node needs encoding decreases. This also indicates that there exists a trade-off between the increment of the network coding capacity and the network
complexity (i.e., the node number and the edge number) in a completely random network.

**Appendix A**

**Proof of Theorem 1**

Given a multicast network as in Theorem 1 and an integer $k > 1$, then can the multicast flow value be larger than or equal to $k$? We denote this problem as “UMMF” (Undirected Graph’s Maximum Multicast Flow) for short. If UMMF is proved to be an NP complete (NPC) problem, then one can deduce that computing the maximum multicast flow $F$ is an NP-Hard problem.

To prove that UMMF is NP-complete, we need to prove the famous NPC problem 3SAT (3-satisfiability) can be transformed to UMMF in polynomial time. Given a 3SAT problem, the Boolean variable set $X = \{x_1, x_2, \ldots, x_n, \bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\}$ and the sentence set $C = \{C_1, C_2, \ldots, C_m\}$ [12]. Each sentence contains three Boolean variables in $X$. The 3SAT problem is to determine whether there exists an assignment of the logic values “true” and “false” to the $n$ variables in $X$. Such an assignment is called proper.

We adopt the topology in Fig. 1 to construct a UMMF problem as in Fig. 7, where the edge $(v_1^i, v_2^j)$ corresponds to $x_i$ in $X$, and $C_1, C_2, \ldots, C_m$ correspond to $m$ destinations. If $x_i$ is true, the flow direction on $(v_1^i, v_2^j)$ is $v_2^j \rightarrow v_1^i$. Otherwise, the direction is $v_1^i \leftarrow v_2^j$. For destination $C_j$, if $C_j$ contains $x_i$, $C_j$ is connected to $v_1^i$ and $v_2^j$. Likewise, if $C_j$ contains $\bar{x}_i$, $C_j$ is connected to $v_2^j$ and $v_1^i$. For example, when $C_1 = \{x_1, x_2, x_3\}$, the connections are shown in Fig. 7.

![Fig. 7. An undirected multicast network constructed from 3SAT.](image)

Obviously the maximum multicast flow from $s$ to $C_1, C_2, \ldots, C_m$ is not less than 3 as can be seen from Fig. 4. Each $C_j$ has three variables. Hence there are three paths of the $s \rightarrow u_k^i \rightarrow C_j$ type. Moreover, they are edge disjoint. If this 3SAT has a proper assignment, it requires $F \geq 4$. Otherwise, if $F < 4$, 3SAT has no proper assignment.

If $F \geq 4$, then for each $C_j$, besides the flow coming from $u_k^i$, there must be at least one edge $(v_1^i, C_j)$ (or $(v_2^j, C_j)$) that carries the flow for it. We denote the flow carried by $(v_1^i, C_j)$ (or $(v_2^j, C_j)$) as $f_j$. If $f_j$ comes from $u_1^i$ ($u_2^j$), it is the same as the flow on $(u_1^i, C_j)$ ($(u_1^i, C_j)$). Thus, it cannot afford to have any new information. So $f_j$ must come from $v_2^j$ ($v_1^i$), and hence $x_i = true \ (\bar{x}_i = true)$. Therefore, each $C_j$ contains a variable of “true” value, and 3SAT has a proper assignment. This proves that each 3SAT can be transformed to a UMMF in polynomial time. That is, 3SAT $\propto$ UMMF, and this shows that UMMF is NPC. By the NP theory of the computing complexity, it follows that computing the maximum multicast flow $F$ in the network given in Theorem 1 is an NP-Hard problem.

**Appendix B**

**A greedy Capacity Allocation Algorithm**

For convenience, we let the symbol $f_i$ denote the flow itself and $\hat{f}_i$ denote the flow value. Besides, $\hat{f}_j^m$ represents the flow $f_j$ that occupies the unit capacity $c_q$.

1. Set $i = 0$, encoded flow value $g = 0$.

2. Divide capacity $C$ into $C$ unit parts, $c_1, c_2, \ldots, c_C$.

3. Find out the flow $f_1, f_2, \ldots, f_m$ for each destination.

4. If all $\hat{f}_j = 0$, terminate.

5. Else

   Select $f_j$ s.t. $\hat{f}_j = \max\{\hat{f}_k\}_{k=1}^m$.

6. If $g = 0$

   7. If $C - i \geq \hat{f}_j$

      8. Allot $c_{i+1} \sim c_{i+f_j}$ to flow $f_j$ for $t_j$

      9. For the rest $t_k$

     10. For $c_{i+1} \sim c_{i+f_j}$ that are allotted to $f_j$, if $\hat{f}_j^m$ is also allotted to $t_k$, set $c_q$ to be the forbidden resource of $f_k$, and remove the corresponding flow of $f_k$ that is the same as $\hat{f}_j^m$, let $\hat{f}_k = \hat{f}_k - 1$.

11. Let $i = i + \hat{f}_j$.

12. Else

13. Allot $c_{i+1} \sim c_C$ to $f_j$, then choose $\hat{f}_j = C + i c_q$ from $c_1 \sim c_i$ that are not forbidden resources of $f_j$ to allot to $f_j$. Thus those flows on $c_q$ are encoded, and $g = g + 1$.

14. Perform the operations as (10) and (11).

15. For the rest $t_k$

16. For the encoded $c_q$ that are just allotted to $f_j$, if $\hat{f}_j^m$ is destined to $t_k$ and other flows on $c_q$ are not destined to $t_k$, set $c_q$ to be the forbidden resource of $f_k$, remove the corresponding flow of $f_k$ that is the same as $\hat{f}_j^m$, $\hat{f}_k = \hat{f}_k - 1$.

17. Let $i = C$.

18. Else if $g > 0$

19. Select as many as possible encoded $c_q$ that are not forbidden resources to allot to $f_j$, suppose the number of such $c_q$ is $Q$.

20. If $\hat{f}_j > Q$

21. Select $\hat{f}_j - Q$ non-encoded $c_q$ that are not forbidden resources of $f_k$ to allot to $f_j$.

22. Perform the operation as (16) and (17).

23. Set $\hat{f}_j = 0$, then go to (4).
ACKNOWLEDGEMENT

The authors would like to thank the editor Prof. Yugang Fang and the anonymous reviewers for their suggestions. This work was partially supported by NPS of China No.60572085 and the research foundation of the State Key Lab. on Mobile Communications of Southeast University, China.

REFERENCES


Khaled Ben Letaief (S’85–M’86–SM’97–F’03) received the B.S degree with distinction in Electrical Engineering from Purdue University at West Lafayette, Indiana, USA, in December 1984. He received the MS and Ph.D. Degrees in Electrical Engineering from Purdue University, in August 1986, and May 1990, respectively. From January 1985 and as a Graduate Instructor in the School of Electrical Engineering at Purdue University, he has taught courses in communications and electronics.

From 1990 to 1993, he was a faculty member at the University of Melbourne, Australia. Since 1993, he has been with the Hong Kong University of Science & Technology where he is currently Chair Professor and Head of the Electronic and Computer Engineering Department. He is also the Director of the Hong Kong Telecom Institute of Information Technology. His current research interests include wireless and mobile networks, Broadband wireless access, OFDM, CDMA, Cooperative networks, Cognitive radio, MIMO, and Beyond 3G Systems. In these areas, he has published over 300 journal and conference papers and given invited keynote talks as well as courses all over the world.

Dr. Letaief served as consultants for different organizations and is the founding Editor-in-Chief of the IEEE Transactions on Wireless Communications. He has served on the editorial board of other prestigious journals including the IEEE Journal on Selected Areas in Communications - Wireless Series (as Editor-in-Chief). He has been involved in organizing a number of major international conferences and events. These include serving as the Co-Technical Program Chair of the 2004 IEEE International Conference on Communications, Circuits and Systems, ICCCS’04; General Chair of the 2007 IEEE Wireless Communications and Networking Conference, WCNC’07; as well as the Technical Program Co-Chair of the 2008 IEEE International Conference on Communication, ICC’08.

In addition to his active research and professional activities, Professor Letaief has been a dedicated teacher committed to excellence in teaching and scholarship. He received the Magoon Teaching Award from Purdue University in 1990; the Teaching Excellence Appreciation Award by the School of Engineering at HKUST (4 times); and the Michael G. Gale Medal for Distinguished Teaching (highest university-wide teaching award and only one recipient/year is honored for his/her contributions).

He is a Fellow of IEEE, an elected member of the IEEE Communications Society Board of Governors, and an IEEE Distinguished lecturer of the IEEE Communications Society. He also served as the Chair of the IEEE Communications Society Technical Committee on Personal Communications and is currently serving as the Chair of the Steering Committee of the IEEE Transactions on Wireless Communications.

Jingyao Zhang received her B.S. degree from Department of Electronic engineering, Tsinghua University, Beijing, China, in 2005. She is currently pursuing her Ph.D. at Department of Electronic Engineering, Tsinghua University, Beijing, China. From Sept. 2006 to now, she is a Research Assistant at Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong. Her research interests include Network Coding, Network Information Theory, Ad Hoc Network and Multiple Access Control etc.

Pingyi Fan received the B.S and M.S. degrees from the Department of Mathematics of Hebei University in 1985 and Nankai University in 1990, respectively, received his Ph.D degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China in 1994. From Aug. 1997 to March 1998, he visited Hong Kong University of Science and Technology as Research Associate. From May, 1998 to Oct. 1999, he visited University of Delaware, U.S.A. as a research fellow. In March 2005, he visited NICT of Japan as visiting Professor.

From June. 2005 to July. 2005, Aug. 2006 to Sept. 2006, he visited Hong Kong University of Science and Technology. He was promoted as full Professor at Tsinghua University in 2002.

Dr. Fan is a member of IEEE and an oversea member of IEICE. He has attended to organize many international conferences including as TPC Chair of International Symposium of Multi-Dimensional Mobile Computing 2004 (MDMC’04) and TPC member of IEEE ICC2005 etc. He is currently an editor of IEEE Transactions on Wireless Communications, Inderscience International Journal of Ad Hoc and Ubiquitous Computing and Wiley Journal of Wireless Communication and Mobile Computing. He is also a reviewer of more than 14 international journals including 10 IEEE journals and three EURASIP journals. His main research interests include B3G technology in wireless communications such as MIMO, OFDM, multicarrier CDMA, space-time coding, and LDPC design etc., network coding, network information theory and cross layer design, etc.