IEICE TRANS. COMMUN., VOL.E92–B, NO.3 MARCH 2009

PAPER Special Section on Ad Hoc and Mesh Networking for Next Generation Access Systems

A More Efficient COPE Architecture for Network Coding in Multihop Wireless Networks

Kaikai CHI[†a)], **Xiaohong JIANG**[†b)], *Nonmembers*, *and* **Susumu HORIGUCHI**[†c)], *Member*

SUMMARY Recently, a promising packet forwarding architecture COPE was proposed to essentially improve the throughput of multihop wireless networks, where each network node can intelligently encode multiple packets together and forward them in a single transmission. However, COPE is still in its infancy and has the following limitations: (1) COPE adopts the FIFO packet scheduling and thus does not provide different priorities for different types of packets. (2) COPE simply classifies all packets destined to the same nexthop into small-size or large-size virtual queues and examines only the head packet of each virtual queue to find coding solutions. Such a queueing structure will lose some potential coding opportunities, because among packets destined to the same nexthop at most two packets (the head packets of small-size and large-size queues) will be examined in the coding process, regardless of the number of flows. (3) The coding algorithm adopted in COPE is fast but cannot always find good solutions. In order to address the above limitations, in this paper we first present a new queueing structure for COPE, which can provide more potential coding opportunities, and then propose a new packet scheduling algorithm for this queueing structure to assign different priorities to different types of packets. Finally, we propose an efficient coding algorithm to find appropriate packets for coding. Simulation results demonstrate that this new COPE architecture can further greatly improve the node transmission efficiency.

key words: *multihop wireless networks, network coding, COPE architecture, packet coding algorithm, transmission efficiency improvement*

1. Introduction

One of the significant problems of multihop wireless networks is that their current implementations suffer from a severe throughput limitation and do not scale well as the number of network nodes increases [1]–[3]. To improve network throughput, the promising network coding technique [4] has been proposed for network nodes to mix data and send the resulting data.

The basic idea of network coding in wireless networks is quite simple and can be illustrated using the scenario in Fig. 1 (from Wu et al. [5]), where node A wants to send packet P_1 to node B and node B wants to send packet P_2 to node A with the help of intermediate node R. Assume node R has received P_1 and P_2. In traditional transmission way, node R transmits P_1 and P_2 separately. However, node R can XOR P_1 and P_2 together and broadcast $P_1 \oplus P_2$. Upon receiving $P_1 \oplus P_2$, node A can decode P_2 by $P_2 = P_1 \oplus (P_1 \oplus P_2)$. Similarly, node B can decode P_1 by $P_1 = P_2 \oplus (P_1 \oplus P_2)$.

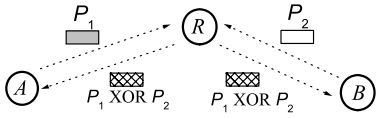


Fig. 1 A simple scenario of wirelesses network coding.

Therefore, with the network coding function, node R can forward two packets in a single packet transmission and its transmission efficiency is improved by 100% when P_1 and P_2 have the same size.

Following the study of the above basic scenario by Wu et al. [5], recently, Katti et al. proposed the first practical network coding-based packet forwarding architecture (called COPE) to essentially improve the network throughput of multihop wireless networks [6]. In COPE, each node can opportunistically overhear and store those native packets transmitted by its neighbors, which are not addressed to itself. Each node can intelligently encode (XOR) multiple packets destined to different nexthops such that multiple packets can be forwarded in a single transmission, resulting in a significant bandwidth saving. Since the proposal of promising COPE architecture, some efforts have been made to theoretically evaluate the performance of COPE-type wireless network coding [7]–[9]. In addition, it has been shown that the optimal coding problem in COPE is NP-complete [10].

Although COPE has demonstrated its capability of improving the network throughput [6], it is still in its infancy and has the following limitations: (1) COPE adopts the FIFO packet scheduler and thus does not enforce different priorities to different types of packets, like routing control packets, voice packets, best-effort packets, etc. (2) COPE simply classifies all packets destined to the same nexthop into small-size or large-size virtual queues and examines only the head packet of each virtual queue to find coding solutions. Such a queueing structure will lose some potential coding opportunities, because among packets destined to the same nexthop at most two packets (the head packets of small-size and large-size queues) will be examined in the coding process, regardless of the number of flows. (3) The coding algorithm adopted in COPE, which finds appropriate packets for coding, is fast but cannot always find good solutions. In order to address the above limitations, in this paper we first present a new queueing structure for COPE, which can provide more potential coding opportunities, and then propose a new packet scheduling algorithm for this queue-

Manuscript received June 15, 2008.
Manuscript revised October 20, 2008.

[†]The authors are with the Graduate School of Information Sciences, Tohoku University, Sendai-shi, 980-8579 Japan.
a) E-mail: kai-chi@ecei.tohoku.ac.jp
b) E-mail: jiang@ecei.tohoku.ac.jp
c) E-mail: susumu@ecei.tohoku.ac.jp

DOI: 10.1587/transcom.E92.B.766

ing structure to guarantee different priorities for different types of packets. Finally, we propose an efficient coding algorithm to find appropriate packets for coding.

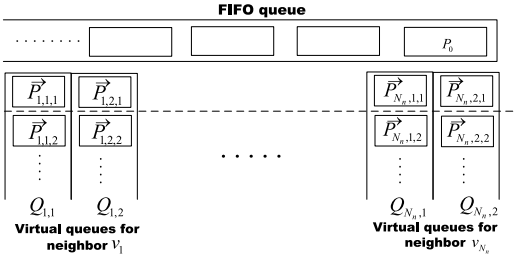
The rest of this paper is organized as follows. In Sect. 2, we briefly review the COPE architecture and describe its limitations. Section 3 presents a new packet queueing structure and a new packet scheduling algorithm. In Sect. 4, we propose an efficient packet coding algorithm. Simulation results are presented in Sect. 5. Finally, Sect. 6 concludes this paper.

2. Overview of COPE

2.1 COPE Architecture

In a COPE-based network [6], a node maintains one FIFO queue (called output queue) and also maintains for each neighbor v_i two virtual queues $Q_{i,1}$ and $Q_{i,2}$ (one for small packets whose sizes are smaller than 100 bytes and another for large packets. See Fig. 2(a)). In addition to these queues, each node also maintains a table, whose entry $\theta_{m,n}$ indicates the probability that neighbor v_m possesses packet P_n at the current time, as illustrated in Fig. 2(b). We refer to the probability $\theta_{m,n}$ as *packet possession indicator* in this paper.

The COPE works as follows. Each node always snoops on all communications over the wireless medium. On one hand, when a node overhears a packet being delivered to another node, it will store the overheard packet in its memory for a limited period (say 0.5 s). On the other hand, when a node successfully receives a native packet or retrieves a native packet from an encoded packet delivered to it, if it is the ultimate destination of this native packet, it delivers the packet to the higher layers of the network stack; otherwise, it first adds this native packet to the output queue, then adds a pointer (pointing to this packet in the output queue) to



(a) Queues inside a network node.

	P_0	P_1	P_2	...
v_1	$\theta_{1,0}$	$\theta_{1,1}$	$\theta_{1,2}$	...
v_2	$\theta_{2,0}$	$\theta_{2,1}$	$\theta_{2,2}$	...
v_3	$\theta_{3,0}$	$\theta_{3,1}$	$\theta_{3,2}$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
v_{N_n}	$\theta_{N_n,0}$	$\theta_{N_n,1}$	$\theta_{N_n,2}$	...

(b) Table of packet possession indicators.

Fig. 2 The data inside a COPE-based network node with N_n neighbors.

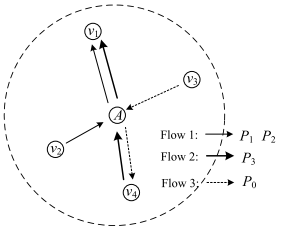
the appropriate virtual queue based on the packet's nexthop and size, and finally updates the hash table by including the probabilities that its neighbors possess this native packet. In addition to overhearing and receiving packets, each node also broadcasts *reception reports* to inform its neighbors the packets it possesses by annotating the data packets or by special control packets. Due to different reasons like packet loss and severe congestion, a node cannot solely rely on reception reports to decide which packets its neighbors possess and thus it may need to estimate the probability that a neighbor possesses a particular packet. If a node learns from reception reports that neighbor v_m possesses packet P_n, then $\theta_{m,n} = 1$. Otherwise, it will estimate the value of $\theta_{m,n}$.

The *packet coding algorithm* inside COPE makes coding decision in the following way. The COPE first dequeues the head packet P_0 of the output queue, and then checks one by one the head packets of virtual queues with the same packet size as P_0 to find appropriate packets to encode with P_0. After exhausting the head packets of the same size as P_0, COPE then checks one by one the head packets of virtual queues of another size. The following rule is adopted to determine if a packet P_{i_n} is feasible to further encode with the currently encoded packet. Suppose we have already decided to XOR n packets $P_0 \oplus P_{i_1} \oplus \cdots \oplus P_{i_{n-1}}$ together, and are considering XOR-ing the $(n+1)$'th packet P_{i_n} with them. The packet coding $P_0 \oplus P_{i_1} \oplus \cdots \oplus P_{i_n}$ is *feasible* only if the following constraint, namely *probability threshold* (PT) constraint, is satisfied: each nexthop to whom a packet $P_i \in \{P_0, P_{i_1}, \ldots, P_{i_n}\}$ is headed can decode its packet P_i with the probability greater than a threshold G (in default, $G = 0.8$ in COPE [6]).

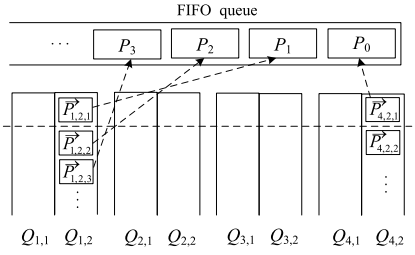
2.2 Limitations of the Available Queueing Structure

The current queueing structure of COPE is quite easy to maintain. However, it has the following two limitations: (1) In multihop wireless networks, it is quite necessary to give priority to some special types of packets (like routing-used control packets) over data packets [11]. Additionally, it is also necessary to set priorities among data packets. Although the FIFO scheduler is trivial to implement, it cannot satisfy this QoS requirement and it also allows rogue flows to capture an arbitrary fraction of the output bandwidth. (2) We should notice that under the condition that no packet reordering is allowed, theoretically all the oldest packets of distinct flows[†] have the potential to code together for throughput improvement. However, the current structure cannot fully explore this potential, because among those packets to the same neighbor at most two packets (the heads of two virtual queues) can be the candidate packets for encoding with P_0. More specifically, when more than one flow with small packets (or large packets) are routed to the same nexthop, only one oldest packet can locate at the head of the virtual queue (i.e., serves as the candidate

[†]Inside a node, the oldest packet of a flow is the firstly arrived packet among all the stored packets of this flow.



(a) Flows passing through A.



(b) FIFO queue and virtual queues inside A.

Fig. 3 Limitation illustration of the current queueing structure.

packet). Therefore, this structure will significantly limit the potential coding opportunities.

To illustrate the limitation of current packet size-based queue structure, we consider a simple example shown in Fig. 3. In this example, Flows 1 and 2 with large-size packets are passing through node A and going to neighbor v_1, while Flow 3 also with large-size packets is passing through node A and going to neighbor v_4. Then, all packets of Flow 1 and 2 will be queued in the same virtual queue $Q_{1,2}$, as shown in Fig. 3(b). Suppose the coding $P_0 \oplus P_1$ is infeasible and the coding $P_0 \oplus P_3$ is feasible. During the search for a feasible coding solution, however, the node A will only check the feasibility of coding $P_0 \oplus P_1$, without the consideration of $P_0 \oplus P_3$. Finally, P_0 will be transmitted alone, resulting in the loss of coding opportunity $P_0 \oplus P_3$.

To address the above two limitations of the available queueing structure in COPE, we propose a new queueing structure and a corresponding packet scheduling algorithm in Sect. 3.

2.3 Limitations of the Available Coding Algorithm

In the original literature of COPE [6], there is no metric available for quantitatively measuring the "goodness" of a coding solution. Here, we introduce a metric for such purpose.

For a coding solution $P_0 \oplus \cdots \oplus P_L$ ($L \geq 0$), let γ be the ratio of the expected number of successfully decoded bytes (after this encoded packet is transmitted) to the size of encoded packet in byte, i.e.,

$$\gamma = \frac{p_0^r \cdot p_0^d \cdot l_0 + p_1^r \cdot p_1^d \cdot l_1 + \cdots + p_L^r \cdot p_L^d \cdot l_L}{\max_{0 \leq i \leq L} l_i}, \qquad (1)$$

where l_k is the size of packet P_k in byte, p_k^r is the probability that the encoded packet can be successfully received by the intended nexthop of P_k, and p_k^d is the probability that the encoded packet can be decoded by the intended nexthop of P_k, $k = 0, \ldots, L$.

The size of encoded packet $P_0 \oplus \cdots \oplus P_L$ is approximately equal to the size of the largest packet[†]. If this encoded packet is transmitted, it is expected that in total $\sum_{i=0}^{L} p_i^r \cdot p_i^d \cdot l_i$ bytes will be successfully decoded at nexthops. Thus, this metric accurately reflects the transmission efficiency improvement that can be achieved during the transmission period of encoded packet. By definition, we can know that $0 \leq \gamma < L + 1$. According to this metric, we can classify different coding solutions into the following three categories. a) $\gamma < p_0^r$: Node's transmission efficiency is lower than that of the non-coding transmission. b) $\gamma = p_0^r$: Node's transmission efficiency keeps unchanged, i.e., it is same as that of transmitting P_0 alone. c) $\gamma > p_0^r$: Node's transmission efficiency is improved in comparison with the non-coding transmission.

The available coding algorithm does not take packet size and delivery ratio into account when searching for the coding solutions, and thus has the following limitations:

1) It skips all infeasible coding solutions, which may have large γ;
2) Many potentially good coding solutions will not be checked (After finding a feasible solution encoding k native packets, the algorithm will stop checking those unchecked solutions which encode k native packets, and attempt to find another native packet to code with the current k native packets.).
3) It may obtain a feasible coding solution which has a small γ. For example, the γ will be small when p_i^r's are small.

Due to the above severe limitations, there usually exist much better coding solutions than the one obtained by this algorithm. We will present an efficient coding algorithm in Sect. 4.

3. Packet Queueing and Scheduling

In this section, we present a new packet queueing structure and also a packet scheduling algorithm.

3.1 Packet Queueing

Rather than queues all packets in a single queue, the new queueing structure is to maintain a dedicated FIFO queue Q_0, called *control queue*, for some special packets (like routing control packets) and maintain a FIFO queue Q_i for each active flow f_i passing through the current node, $i \geq 1$, as shown in Fig. 4.

Such a queueing structure can provide more potential coding opportunities. Let us still consider the example in Fig. 3. With the proposed queueing structure, P_3 will be at the head of the queue maintained for Flow 2 and thus the

[†]In each encoded packet's header, several symbols are used for recording the number of native packets XOR-ed together, IDs of native packets, etc.

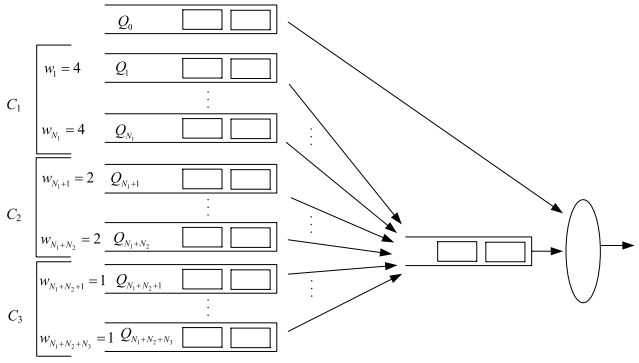


Fig. 4 Flow-oriented queueing structure.

coding algorithm can find the feasible coding solution $P_0 \oplus P_3$. This example indicates that the new queueing structure increases candidate packets and consequently increases the coding opportunities.

Furthermore, this new queueing structure enables us to not only easily give higher priority to those special packets than to data packets, but also easily assign weights or priorities among data packets.

3.2 Packet Scheduling

With the above queueing structure, we proceed to specify how to assign transmission chances to data flows.

When a network node obtains a transmission chance from the MAC layer, its packet scheduler first checks whether the control queue is nonempty. If so, it will dequeue the packet at the head of control queue and transmit it alone (encoding it with other packets will decrease the probability of its successful delivery). In other words, data packets have no chance for transmission until there is no any packet in the control queue. Note that since packets in control queue only account for a small percentage of all buffered packets, giving priority to these packets almost does not affect the end-to-end delay of data packets [11].

In the following, we introduce how to schedule data packets. Similar to IEEE 802.11e, In our scheduling algorithm, traffic flows are also separated into the following three classes: flows of voice packets, flows of video packets and flows of best-effort packets, denoted by F_1, F_2 and F_3, respectively, as shown in Fig. 4. Let N_i denote the number of flows belonging to class F_i for $i = 1, 2$ and 3, and let N denote the total number of active flows (i.e. $N = \sum_{i=1}^{3} N_i$). To achieve the target of giving higher priority to voice and video packets than to best-effort packets, we allocate larger weights to voice and video flows than to best-effort traffic flows. Denote by w_i the weight of flow f_i, and let W be $W = \sum_{i=1}^{N} w_i$. We expect that the percentage of transmission times assigned to flow f_i is approximately equal to w_i/W. Now an obvious problem arising is the appropriate value setting of w_i. In the IEEE 802.11e standard which supports multimedia applications such as voice and video over the IEEE 802.11 WLANs, by default, the contention window (CW) of best effort traffic is four times as large as voice packets' CW and two times as large as video packets' CW. Thus, a reasonable setting of w_i is as follows: $w_i = 4$ for each voice flow f_i, $w_j = 2$ for each video flow f_j and $w_k = 1$ for each best-effort traffic flow f_k. Note that, upon the specific requirement we can also separate packets into classes in other ways and set their respective weights. For example, among best-effort flows, web surfing can have larger weight than FTP and email applications.

With the above specifications, we schedule packets in a similar manner as round robin scheduling [13]. In order to explain the scheduling algorithm, we first clarify two concepts: *small-round exploring* and *large-round exploring*. A round of exploring N flows one by one is called a small-round exploring, and the conduction of $\max_{1\le i\le N} w_i$ times of the small-round exploring is called a large-round exploring. For the above setting, $\max_{1\le i\le N} w_i = 4$. Let I denote the ID of the flow which will be serviced at the current transmission time. In this scheduler, parameters R_i ($1 \le i \le N$) are adopted to determine whether the scheduler will service a flow or skip over it. When starting a new large-round exploring, for each i initialize R_i as $R_i = w_i$ and set I to 1. R_i represents the number of times flow f_i needs to be serviced during the remaining services of the current large-round exploring. When a node obtains a transmission chance and the control queue is empty, the scheduler dequeues the packet at the head of Q_I and select by the coding algorithm appropriate packets to code with P_I. One important point we should notice is that by using network coding, multiple native packets can be forwarded by the transmission of an encoded packet. To achieve the target that the percentage of transmission times assigned to flow f_i is approximately equal to w_i/W, for each successfully decoded native packet P_i let $R_i = R_i - 1$. If P_I is successfully forwarded in the current transmission, conduct $I = I + 1$ until $R_I > 0$. Otherwise, keep I unchanged.

Now the scheduling algorithm is summarized as follows.

Scheduler

Dequeue the packet P_I at the head of Q_I.
Find appropriate packets to code with P_I by the coding algorithm and transmit the encoded packet.
for each successfully forwarded P_i **do**
  $R_i = R_i - 1$
end for
if P_I is successfully forwarded **then**
  $I = I + 1$
  if $I > N$ **then**
    $I = 1$
  end if
end if
if all R_i are equal to zero **do**
  for $i = 1$ to N **do**
    $R_i = w_i$
  end for
  $I = 1$
else if $R_I = 0$ **do**
  while $R_I = 0$ **do**
    $I = I + 1$
  end while
end if

Similar to the round robin scheduler, compared to the

FIFO scheduler in COPE, such a scheduler has two important advantages: first, it prevents a rogue source from arbitrarily increasing its share of the bandwidth; second, it satisfies the QoS requirement of multihop wireless networks.

4. Efficient Packet Coding Algorithm

To take full advantage of the coding opportunities provided by the new queueing structure, in this section, we present a more efficient coding algorithm than the available one in COPE.

As discussed previously, the available coding algorithm has several limitations which may lead to the obtaining of a not-so-good coding solution in the case there exist good coding solutions. However, it is possible for us to have a very efficient search for good coding solutions, due to the following good properties:

P1: Good coding solutions usually have high decoding probabilities, so they are very likely to satisfy the PT constraint (i.e. be feasible coding solutions). Therefore, we are able to greatly shrink the search space by searching for a good coding solution only among the feasible solutions.

P2: In most cases, encoding too many packets together will result in small decoding probabilities. We can achieve good performance by encoding not more than a given number of native packets (say 4) in all cases, as indicated in [6].

Based on the above properties, the goal of our coding algorithm is to find the best coding solution (with the largest γ) only among feasible coding solutions which encode at most N_{max} native packets. The appropriate value of N_{max} will be determined by virtue of simulation results. To describe this new coding algorithm, we first introduce a special type of directed graph, called *coding graph*.

Definition 2: (Coding Graph) Given knowledge (like packet size) of packet P_I being served by the packet scheduler and knowledge of all packets P_i's at the heads of queues, construct a corresponding coding graph $\mathcal{G}(\mathcal{V}, \mathcal{A})$ as follows:

- The vertex set $\mathcal{V}$ of $\mathcal{G}$ is defined as $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$, where node v_i corresponds to packet P_i. Assign two weights $s_i = l_i$ and $z_i = p_i^r$ to each node v_i.
- The arc set $\mathcal{A}$ of $\mathcal{G}$ is defined as: for each v_i ($i \neq I$) satisfying $\theta_{N(P_i),I} > G$ and $\theta_{N(P_I),i} > G$, where $N(P_i)$ represents the nexthop ID of P_i, there are an arc (v_I, v_i) with weight $p_{(v_I,v_i)} = \theta_{N(P_i),I}$ and an arc (v_i, v_I) with weight $p_{(v_i,v_I)} = \theta_{N(P_I),i}$; between any two vertexes v_i ($i \neq I$) and v_j ($j \neq I$) from which there are arcs to v_I, if $\theta_{N(P_j),i} > G$ and $\theta_{N(P_i),j} > G$, there are an arc (v_i, v_j) with weight $p_{(v_i,v_j)} = \theta_{N(P_j),i}$ and an arc (v_j, v_i) with weight $p_{(v_j,v_i)} = \theta_{N(P_i),j}$.

For a subgraph of $\mathcal{G}$, call it a feasible coding subgraph if:

(a) it contains v_I;

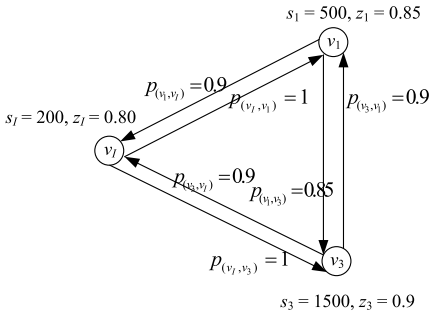


Fig. 5 An example of a feasible coding graph for $G = 0.8$.

(b) between any two different nodes u_i and u_j in it, these are arcs (u_i, u_j) and (u_j, u_i);

(c) for each node of this subgraph, the product of weights of all arcs entering this node is larger than G.

Note that a feasible coding subgraph $\mathcal{G}_f(\mathcal{V}_f, \mathcal{A}_f)$ in $\mathcal{G}$ corresponds to a feasible coding solution. Figure 5 shows a simple feasible coding subgraph, whose corresponding feasible coding solution is $P_I \oplus P_1 \oplus P_3$.

Let the weight $W(\mathcal{G}_f)$ of a feasible coding subgraph $\mathcal{G}_f$ be

$$W(\mathcal{G}_f) = \frac{\sum_{v_i \in V_f} \left(s_i \cdot z_i \cdot \prod_{(v_j,v_i) \in \mathcal{A}_f} p_{(v_j,v_i)} \right)}{\max_{v_i \in \mathcal{V}_f} s_i}. \quad (2)$$

Clearly, the weight of a feasible coding subgraph is equal to the γ of the corresponding feasible coding solution. Formally, the coding algorithm is as follows.

Packet Coding Algorithm

Input:
Value of I and size of each head packet P_i ($1 \leq i \leq N$)
Values of all $\theta_{m,n}$'s ($0 \leq n \leq N$)
Packet delivery ratio p_k^r ($0 \leq k \leq N$)
Procedure
Based on the input, construct the corresponding coding graph $\mathcal{G}$.
$W_{max} = 0$.
for $k = 2$ to $min\{N_{max}$, node number of $\mathcal{G}\}$ **do**
  for each subgraph $\mathcal{G}'$ containing v_I and also $k - 1$ other vertexes **do**
    if $\mathcal{G}'$ is feasible **do**
      if $W(\mathcal{G}') > W_{max}$ **do**
        $W_{max} = W(\mathcal{G}')$
      end if
    end if
  end for
end for
Exit: Return the feasible subgraph which includes node v_I and has the largest weight.

This new coding algorithm takes $O(N^{N_{max}-1})$ time, which is quite fast when N_{max} is small. Simulation results in the next section will demonstrate that setting N_{max} to 3 can achieve good enough performance. Thus, this algorithm only takes $O(N^2)$ time when $N_{max} = 3$.

5. Simulation Results and Analysis

In this section, we investigate how much the node transmis-

sion efficiency can be further improved by adopting the proposed queueing structure and coding algorithm in COPE, as compared with the original COPE.

Since the γ defined previously is used for measuring the short-term (one packet transmission period) transmission efficiency improvement, we define here a new metric to measure the long-term performance in terms of the node transmission efficiency. Let E_c and E_{nc} represent the average number of bytes delivered to neighbors per transmitted byte when using coding-based transmission and using non-coding (traditional) transmission, respectively. Using non-coding transmission, network nodes transmit native packets and suffer packet loss. Thus $E_{nc} < 1$. However, in the COPE-based networks, network nodes can forward multiple packets in a single packet transmission, so E_c can be larger than one there. Then we define the *node transmission efficiency improvement* (NTEI) ρ as

$$\rho = E_c/E_{nc}. \tag{3}$$

This metric clearly reflects the improvement in the node transmission efficiency, independent of the adopted physical layer protocol (i.e. the bit-rate) and the MAC layer protocol.

5.1 Simulation Setting

The performance evaluation is conducted on network configurations randomly generated as follows. (a) Random topology generation: First, place relay node A at coordinate (0, 0). Then N_n neighbors are randomly and independently distributed within the transmission range of unit one. Each generated topology consists of one transmission node and several neighbors. The transmission node continuously transmits packets (which are native packets when using the non-coding transmission way and are encoded packets when adopting network coding) and the neighbors receive the packets. The E_c and E_{nc} are the ratios of the total number of successfully delivered bytes to the total number of transmitted bytes when using the coding-based transmission way and the non-coding transmission way, respectively. (b) Due to the small percentage of special packets link control packets, only data packets are considered in the simulation. For each data flow, randomly select two neighbors X and Y. If their distance $d(X, Y) \leq 1$, randomly select two neighbors again until $d(X, Y) > 1$[†]. Then this flow will be routed through $X \rightarrow A \rightarrow Y$. Each best-effort TCP flow comprises of a forward flow of data packets and a reverse flow of ACK packets with size 40 bytes. The data packet size of a flow remains unchanged and follows the packet size distribution presented in [14]. In addition, we consider the case that the flows are infinite and steady, and each flow always has packets in the output queue.

For wireless channels we adopt the Rayleigh block fading model and approximate the packet error rate of a channel with the probability that the instantaneous received SNR is smaller than a fixed threshold γ_T [15]. Then packet possession indicator $\theta_{m,n}$ is estimated based on the following model proposed in [16]: $\theta_{m,n} = \exp(-\frac{\gamma_T}{K} d^{\alpha})$, where d is the link distance, α is the path loss exponent and K is a constant depending on the transmitting power, the antenna gain, etc. The path loss exponent α is set to 4, and γ_T/K is set to 0.2, achieving a delivery ratio about 0.82 between two nodes with unit distance.

For each setting of the numbers of flows and neighbors, we generate 5000 random configurations. For each configuration, we simulate the packet transmissions by using the non-coding transmission, the original COPE-based transmission and the improved COPE-based transmission, respectively. The observed NTEIs of the original COPE and the improved COPE are finally averaged over 5000 configurations.

5.2 Shortcoming of Probability Threshold Constraint

The available coding algorithm in COPE aims to encode as many as possible native packets together while satisfying the PT constraint. However, the PT constraint only considers the probabilities of successful decoding at nexthops, but does not take into account sizes of native packets and the link delivery ratios of those links from the delay node to nexthops. Thus, a feasible coding solution encoding many packets does not necessarily have a large γ.

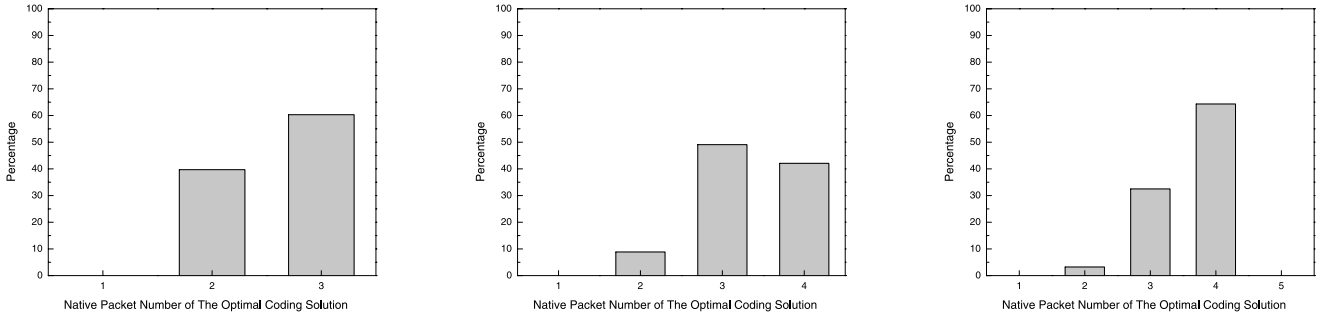
For a coding problem, let N_m be the maximum number native packets which can be encoded together while satisfying the PT constraint. Figure 6 shows the distribution of native packet number of the optimal coding solution, among the cases with the same N_m. We can see that although at most N_m packets can be encoded together while satisfying the PT constraint, the optimal coding solutions are often some solutions which encodes less than N_m. For example, for those cases with $N_m = 5$, all the optimal coding solutions encode less than five native packets. Therefore, the PT constraint is not good enough as a metric for measuring the "goodness" of a coding solution. It is quite necessary to take into account sizes of native packets and the link delivery ratios of those links from the delay to nexthops, as shown in Equation (1).

5.3 NTEI versus Maximum Number of Packets Allowed to Encode Together

Now we will investigate the appropriate setting of the maximum number N_{max} of packets allowed to encode together in the proposed coding algorithm. Figure 7 shows the average NTEI under different values of N_{max}. We can observe that compared to the setting $N_{max} = 2$, the setting $N_{max} = 3$ leads to a much larger average NTEI. However, setting N_{max} to 4 or a larger value only very slightly increases the average NTEI. Therefore, we can conclude that setting N_{max} to 3 can achieve good enough performance.

In order to clearly understand this performance characteristic, we further examine in great detail the distribution of

[†]Traffic between in-range nodes does not need to be forwarded by the relay.

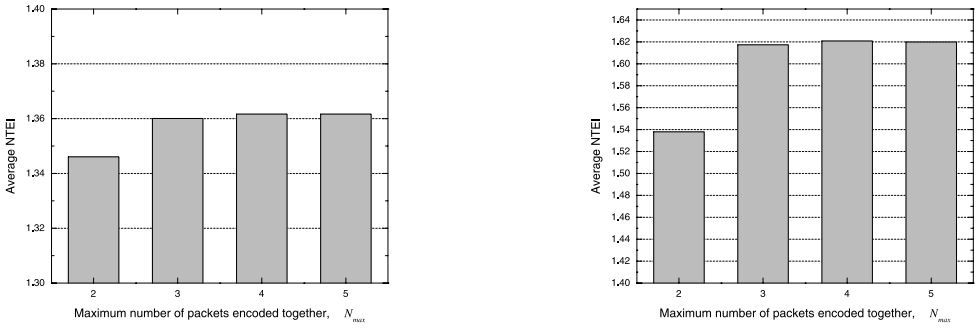


(a) Distribution among those cases with $N_m = 3$.

(b) Distribution among those cases with $N_m = 4$.

(c) Distribution among those cases with $N_m = 5$.

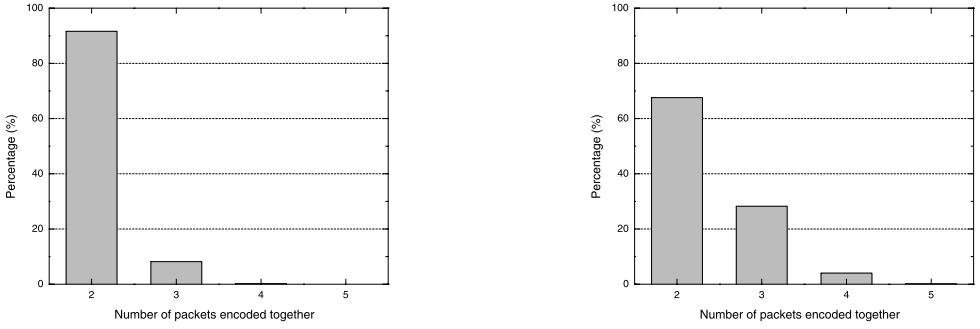
Fig. 6 Distribution of the number of native packets in the optimal coding solutions.



(a) Average NTEI of nodes with normal traffic load and a normal number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 2$ and $N_n = 4$)

(b) Average NTEI of nodes with heavy traffic load and a large number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 5$ and $N_n = 7$)

Fig. 7 NTEI versus the maximum number of packets allowed to encode together.



(a) At nodes with normal traffic load and a normal number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 2$ and $N_n = 4$)

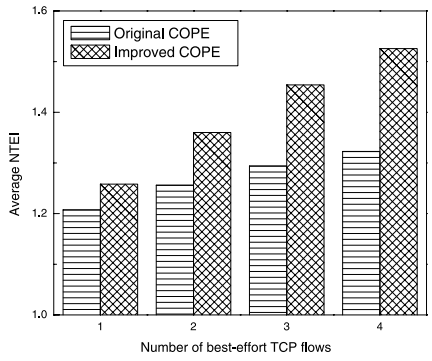
(b) At nodes with heavy traffic load and a large number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 5$ and $N_n = 7$)

Fig. 8 Distribution of number of packets coded together.

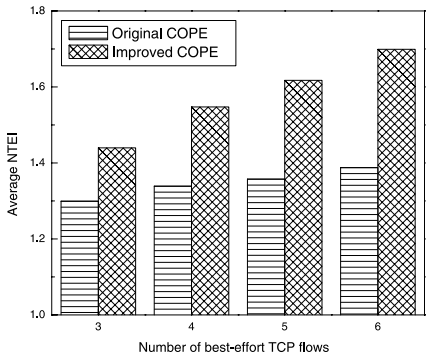
the number of native packets encoded together in Fig. 8. We can see that it is very rarely happen to encoded four or more packets together. This is easy to understand. Let us take the case of encoding four native packets as an example. To encode four packets together, each one of four nexthops of the encoded packet needs to possess other three packets except the packet destined to it. This condition is so strict that it can be rarely satisfied. Due to the low probability of encoding four or more packets, setting N_{max} to 3 can achieve good enough performance and also lead to a low computational complexity of the coding algorithm.

5.4 Comparison between the Original COPE and Improved COPE

In this subsection, we investigate the improvement achieved by adopting the new queueing structure and new coding algorithm. Figure 9 shows the average NTEI achieved by the

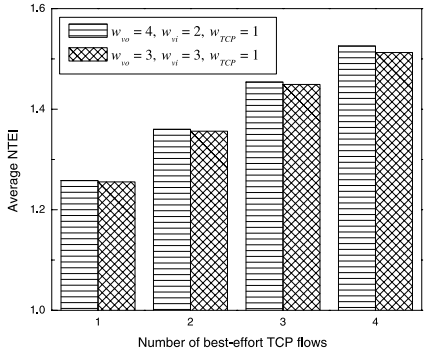


(a) At nodes with four neighbors. ($N_1 = 1, N_2 = 1$)

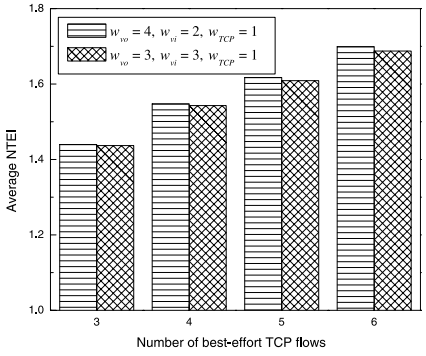


(b) At nodes with seven neighbors. ($N_1 = 1, N_2 = 1$)

Fig. 9 Comparison between the original COPE and improved COPE.



(a) At nodes with four neighbors. ($N_1 = 1, N_2 = 1$)



(b) At nodes with seven neighbors. ($N_1 = 1, N_2 = 1$)

Fig. 10 NTEI versus different settings of flow weight.

original COPE and the improved COPE, respectively. We can see that the improved COPE always significantly outperforms the original COPE. For example, the average NTEI of nodes with 7 neighbors, one voice flow, one video flow and four TCP flows, is improved by 15.6%. In addition, the improvement increases as the number of active flows increases. This is because compared to nodes with few active flows, nodes with a lot of active flows have more potential coding opportunities and thus remain larger scope for improvement.

5.5 NTEI under Different Settings of Flow Weight

In the scheduling algorithm, different types of flows are assigned with different weights. Now we will investigate whether the algorithm performance is sensitive to the assignment of flow weight. Figure 10 shows the average NTEI under two different settings of flow weight. We can see that the average NTEI almost keep unchanged under these two settings. The same conclusion can be drawn when other settings are used. Therefore, we can expect that the NTEI will only slightly change when other schedulers like the one in [12] are adopted for the proposed queueing structure.

Table 1 The average solution search time under different numbers of passing flows. $N_1 = 1$, $N_2 = 1$ and $N_n = 7$.

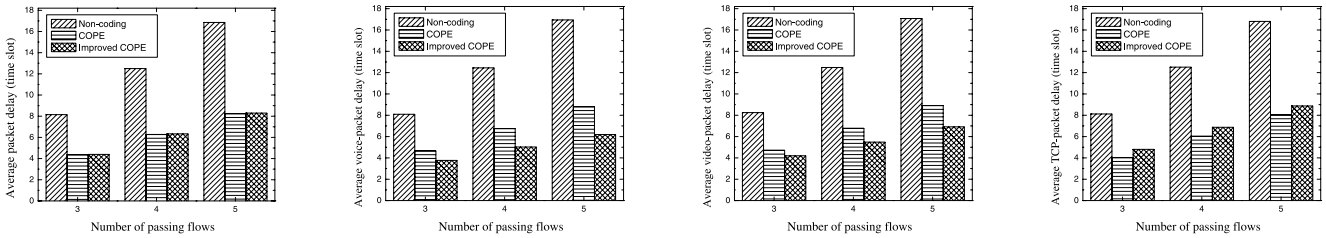
N_3	1	2	3	4	5
Average search time (μs)	2.81	3.18	4.48	5.02	5.80

5.6 Packet Delay

Here we investigate the delay performance of the improved COPE.

First, the storing function will not increase the packet delay. The COPE's storing function at one node is used only to store the overheard packets (not the forwarded packets) for a period in a particular buffer, which is not the buffer for queueing the packets to be forwarded. The packets needing to be forwarded wait in their own queue for getting their transmission chances, just like the current packet forwarding architecture.

Then we investigate the average running time for the search of coding solution. Table 1 shows the average running time for finding a set of packets for coding under different numbers of passing flows. From this table we can see that the solution search time is at the microsecond level



(a) Average delay of all packets.
(b) Average delay of voice packets.
(c) Average delay of video packets.
(d) Average delay of TCP packets.

Fig. 11 Average queueing delays of different schemes. ($N_1 = 1$, $N_2 = 1$, $N_n = 4$.)

and is very small as compared with other parts like queueing delay. As for packet coding (i.e. XOR-ing) and packet decoding, they are linear operations and consume almost neglectable time.

Finally, we compare the queuing delay performance between the non-coding transmission with FIFO buffer, the current COPE-based transmission and our improved COPE-based transmission in the following way: each flow passing through node A has 4 packets in the buffer queue and we simulate the average packet queueing delay during the transmission of these buffered packets. Here we assume that node A continuously transmits packets and each transmission take a time slot of fixed duration. Figure 11 shows the average delay of all packets, the average delay of voice packets, the average delay of video packets and the average delay of TCP packets, respectively. First, we can observe that both the COPE-based transmission and the improved COPE-based transmission greatly outperform the traditional non-coding transmission, because they can much faster deliver buffered packets of a node to this node's neighbors. In addition, from Figs. 11(b) and 11(c) we can see that, compared to the COPE-based transmission, the improved COPE-based transmission leads to a smaller average delay of voice packets and a smaller average delay of video packets. This is because the improved COPE-based transmission gives high priority to the voice packets and video packets, at the cost of slightly increasing the average delay of TCP packets (as shown in Fig. 11(d)).

5.7 The End-to-End Throughput

The node-level transmission efficiency improvement and delay performance improvement shown above can also suggest that the end-to-end throughput will be improved. We can understand this in the following way. Using the network coding technique to forward multiple packets via one packet transmission, is just like using a larger transmission bandwidth to improve the node transmission rate. Our improved COPE can forward more packets per packet transmission than the current COPE by more effectively utilizing the network coding technique. Therefore, the improve COPE can further improve the node-level performance (the transmission rate and packet delay) and consequently will improve the network-level performance.

6. Conclusion

In this paper, we presented for the COPE architecture a new flow-oriented queueing structure which can increase the potential coding opportunities and are convenient for the allocation of priorities to packets, and also proposed a new efficient packet coding algorithm. Rather than adopting FIFO scheduler, allocating priorities to different flows can satisfy the QoS requirement of multihop wireless networks for supporting real-time services such as voice applications. To our knowledge, this is the first time to take the QoS issue into account in the literature of wireless network coding. Simulation results demonstrate that by adopting the new queueing structure and new coding algorithm, COPE can further greatly improve the node transmission efficiency.

As the first step, in this work we investigate the node-level performance improvement by using the new COPE architecture. In the future, it is quite interesting to extend this work by investigating how much the network-level performance (like the end-to-end throughput) can be improved under different workloads, routing protocols, etc.

Acknowledgment

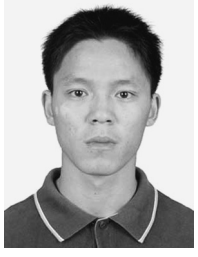
The authors would like to thank the editor and the anonymous reviewers for their many valuable comments which helped to considerably improve the paper.

This work is supported in part by JSPS Grant-In-Aid of Scientific Research (C) 19500050 and (B) 20300022, and the State Key Laboratory for Novel Software Technology, Nanjing University, China.

References

[1] P. Gupta and P.R. Kumar, "The capacity of wireless networks," IEEE Trans. Inf. Theory, vol.46, no.2, pp.388–404, March 2000.
[2] J. Li, C. Blake, D.S.J. De Couto, H.I. Lee, and R. Morris, "Capacity of ad hoc wireless networks," MOBICOM'01, 2001.
[3] J. Jun and M.L. Sichitiu, "The nominal capacity of wireless mesh networks," IEEE Wireless Commun., vol.10, no.5, pp.8–14, Oct. 2003.
[4] R. Ahlswede, N. Cai, S.-Y.R. Li, and R.W. Yeung, "Network information flow," IEEE Trans. Inf. Theory, vol.46, no.4, pp.1204–1216, July 2000.
[5] Y. Wu, P.A. Chou, and S.-Y. Kung, "Information exchange in wireless networks with network coding and physical-layer broadcast,"

CISS'05, 2005.
[6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XOR in the air: Practical wireless network coding," ACM SIGCOMM, Pisa, 2006.
[7] J. Liu, D. Goeckel, and D. Towsley, "Bounds on the gain of network coding and broadcasting in wireless networks," IEEE INFOCOM, 2007.
[8] S. Sengupta, S. Rayanchu, and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," IEEE INFOCOM, 2007.
[9] K. Chi, X. Jiang, and S. Horiguchi, "Network coding opportunity analysis of COPE in multihop wireless networks," IEEE WCNC, Las Vegas, 2008.
[10] K. Chi, X. Jiang, and S. Horiguchi, "A general packet coding scheme for multi-hop wireless networks," Proc. IEEE GLOBECOM, Washington, 2007.
[11] B.G. Chun and M. Baker, "Evaluation of packet scheduling algorithms in mobile ad hoc networks," ACM Mobile Computing and Communications Review, vol.6, no.3, pp.36–49, July 2002.
[12] S. Ramabhadran and J. Pasquale, "Stratified round robin: A low complexity packet scheduler with bandwidth fairness and bounded delay," ACM SIGCOMM'03, 2003.
[13] J. Nagle, "On packet switches with infinite storage," IEEE Trans. Commun., vol.35, no.4, pp.435–438, 1987.
[14] "Internet packet size distributions: Some observations," http://netweb.usc.edu/˜rsinha/pkt-sizes/
[15] M. Zorzi and S. Pupolin, "Optimum transmission ranges in multihop packet radio networks in the presence of fading," IEEE Trans. Commun., vol.43, no.7, pp.2201–2205, July 1995.
[16] H. Yang, K. Wu, and W. Lu, "Cross-layer path configuration for energy-efficient communication over wireless ad hoc networks," Advances in Multimedia, vol.2007, Article ID 19860, 2007.



Kaikai Chi received the B.S. and M.S. degrees from Xidian University, Shaanxi, China, in 2002 and 2005, respectively. He is now a Ph.D. candidate in the Graduate School of Information Sciences, Tohoku University, Sendai, Japan. His current research focuses on the application of network coding in wired networks and wireless networks, respectively, such as the topology design of network coding-based networks and the design of routing algorithms in network coding-capable networks. He is a student member of the IEEE.

Xiaohong Jiang received his B.S., M.S. and Ph.D. degrees in 1989, 1992, and 1999 respectively, all from Xidian University, Xi'an, China. He is currently an Associate Professor in the Department of Computer Science, Graduate School of Information Science, TOHOKU University, Japan. Before joining TOHOKU University, Dr. Jiang was an assistant professor in the Graduate School of Information Science, Japan Advanced Institute of Science and Technology (JAIST), from Oct. 2001 to Jan. 2005. Dr. Jiang was a JSPS (Japan Society for the Promotion of Science) postdoctoral research fellow at JAIST from Oct. 1999–Oct. 2001. He was a research associate in the Department of Electronics and Electrical Engineering, the University of Edinburgh from March 1999–Oct. 1999. Dr. Jiang's research interests include optical switching networks, routers, network coding, WDM networks, VoIP, interconnection networks, IC yield modeling, timing analysis of digital circuits, clock distribution and fault-tolerant technologies for VLSI/WSI. He has published over 130 referred technical papers in these areas. He is a member of IEEE.

Susumu Horiguchi received the B.Eng. the M.Eng. and Ph.D. degrees from Tohoku University in 1976, 1978 and 1981 respectively. He is currently a Full Professor in the Graduate School of Information Sciences, Tohoku University. He was a visiting scientist at the IBM Thomas J. Watson Research Center from 1986 to 1987. He was also a professor in the Graduate School of Information Science, JAIST (Japan Advanced Institute of Science and Technology). He has been involved in organizing international workshops, symposia and conferences sponsored by the IEEE, IEICE, IASTED and IPS. He has published over 150 papers technical papers on optical networks, interconnection networks, parallel algorithms, high performance computer architectures and VLSI/WSI architectures. Prof. Horiguchi is members of IPS and IASTED.