Adaptive Coding Optimization in Wireless Networks: Design and Implementation Aspects

Yi Shi, *Senior Member, IEEE*, Yalin E. Sagduyu, *Member, IEEE*, Junshan Zhang, *Fellow, IEEE*, and Jason H. Li, *Member, IEEE*

***Abstract*—A fundamental challenge in wireless networks is how to handle packet loss due to noise, interference, and dynamic channel effects, especially when there is no per-packet acknowledgement due to additional delay and potential loss of feedback packets. We design and implement a packet coding optimization scheme, applied at the source node, to enhance end-to-end transmission reliability in a lossy multi-hop network. Specifically, each source node transmits a file of packets to its destination node in the network where the end-to-end packet acknowledgement is not readily available because of the possible error or delay effects over multiple hops. By simply retransmitting packets, a source node cannot guarantee innovative packet arrivals at the destination node. Thus, we design an optimization scheme for adaptive packet coding applied at the source node to avoid transmitting redundant packets, thereby significantly improving the throughput, even for the case of a single unicast session. This scheme does not require any knowledge of network topology and can seamlessly operate with any routing or network coding protocol used at the intermediate relay nodes. We analyze the throughput properties of coded transmissions and verify the feasibility of performance gains via simulations. Then, we provide high fidelity emulation testbed results with real radio transmissions over emulated channels to evaluate the throughput gains. Without relying on end-to-end acknowledgment for each packet, we show that the adaptive packet coding optimization scheme can achieve significantly higher throughput than the retransmission scheme in lossy networks with unicast traffic.**

***Index Terms*—Network coding, retransmission, lossy networks, multi-hop networks, throughput, network emulation, testbed.**

I. Introduction

DIFFERENT disruptive effects of noise, interference, dynamic channels, and mobility contribute to end-to-end packet losses in wireless networks. It is challenging to handle packet loss when additional delay and potential loss of feedback packets make per-packet acknowledgement impractical. We consider the design and implementation of reliable communications in a lossy multi-hop network, where end-to-end transmissions may fail on paths from source nodes to destination nodes, and show the throughput benefits of coding optimization applied at source nodes in realistic testbed implementation. In this setup, a packet may be lost due to real-time link or node failure, or may be dropped if there is a (hard delay) deadline for end-to-end packet delivery. Thus to ensure successful end-to-end transmissions in a lossy network, it is necessary to continue transmitting packets until they are successfully received.

We consider a model that every source node transmits a file (a group of packets) to its destination node. We focus on the case where if any packet is lost, the entire transmission for the file fails with zero gain. An example scenario is that this group of packets corresponds to a compressed (e.g., zip) file. The destination node can unzip it only if the entire zip file is received. For the ideal case that there is instantaneous and reliable transport layer end-to-end acknowledgement (ACK) for each packet, a source node can simply retransmit lost packets (without loss of throughput optimality). However, the end-to-end ACK delay in a multi-hop network is at least the round-trip-time (RTT), which grows with the network size. Moreover, an ACK message may also be lost as data packets. Hence, we do not assume that a source node knows exactly which packet is lost.

A simple scheme for reliable transmissions is that a source node keeps retransmitting each packet until a high end-to-end success probability is achieved. Because of the lack of ACKs, it is possible that a packet is retransmitted even if it is already received at the destination. To address this redundancy issue, we implement an adaptive packet coding scheme optimized for reliable transmissions using only end-to-end loss rate estimates rather than per packet feedback. The source node does not transmit original packets but transmits coded packets such that each of them contributes to the recovery of the entire file. We analyze the performance for both schemes. This packet coding scheme has better throughput performance than the simple retransmission scheme for unicast traffic with single or multiple source-destination pairs.

We note that packet coding scheme considered in this paper departs from the traditional network coding scheme, which extends store-and-forward-based routing and provides the general capability to process traffic at relay nodes [1], [2]. It is well known that network coding improves the throughput compared to routing in reliable networks with multicast traffic (where the packet loss probability is negligible if data rate is no more than link capacity) and linear coding is sufficient for single source

Manuscript received December 15, 2014; revised March 16, 2015; accepted May 18, 2015. Date of publication June 3, 2015; date of current version October 8, 2015. This material is based upon work supported by the Air Force Office of Scientific Research under Contracts FA9550-12-C-0037, FA9550-10-C-0026 and FA9550-11-C-0006. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Air Force Office of Scientific Research. The associate editor coordinating the review of this paper and approving it for publication was Q. Li.

Y. Shi, Y. E. Sagduyu, and J. H. Li are with Intelligent Automation Inc., Rockville, MD 20855 USA (e-mail: yshi@i-a-i.com; ysagduyu@i-a-i.com; jli@i-a-i.com).

J. Zhang is with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: Junshan.Zhang@asu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TWC.2015.2441060

multicast [3]. In such reliable networks, network coding can increase the throughput for multiple unicasts but not necessarily for a single unicast session [4].

In this paper, we implement adaptive packet coding and optimization at the source node in a multi-hop network with non-negligible end-to-end packet loss probabilities, where arbitrary routing or network coding operations can be applied at relay nodes. One-hop multicast in a lossy network was studied for error-free ACKs [5], [6] and erroneous ACKs [7], and it was shown that network coding can decrease the packet delay and consequently increase the throughput. Instead, we study the throughput directly in a multi-hop lossy network with unicast traffic. Without end-to-end ACK for each packet, the source node transmits a group of packets, which should be all received at the destination node. Based on the estimated end-to-end packet loss probability, the source can adaptively determine a suitable number of packets in a group (i.e., the original size of packet groups) under a given number of coded packets or a suitable number of coded packets (i.e., the number of transmissions or the delay bound for a file) under a given number of packets in a group such that throughput can be maximized. Through analytical and simulation studies, we show that network coding can improve the throughput even for a single source unicast in this lossy network model.

We also provide emulation testbed results, where we evaluate the performance of adaptive packet coding optimization in a wireless network environment with actual radio transmissions. Network coding schemes have been studied in testbeds with fixed network topologies, e.g., [8]–[11]. In this paper, we use a real-time wireless emulation platform, RFnest [12] (developed by Intelligent Automation, Inc. and offered as a COTS product), to generate dynamic network scenarios and physical channel events. We verify the improved throughput performance by adaptive packet coding optimization under different end-to-end loss events generated with actual radios. We also run real video traffic in the emulation environment and observe that the video quality significantly improves with the adaptive packet coding optimization scheme in consistence with the measured throughput increase.

The remaining of this paper is organized as follows. Section II presents the system model. A simple retransmission scheme is given and is analyzed in Section III. In Section IV, we design an adaptive scheme for packet coding optimization, discuss its throughput benefits, and analyze its performance. Section V evaluates the performance through simulations and emulations. Section VI concludes this paper.

II. System Model

We consider a multi-hop lossy wireless network that provides best-effort services, i.e., no strict guarantee on the end-to-end packet delivery. There are random end-to-end packet loss events (on the path from a source to its destination), caused by either node failure, link failure, or deadline-based packet expiration. We do not restrict the model to a particular network-layer operation, which can be based on either routing or network coding to route packets over the network. In addition, we do not require any prior knowledge on node or link error probabilities.

A source node wants to transmit a file of m packets to the corresponding destination node over this lossy network. We consider the case that if any packet is lost, the entire transmission for the file fails with zero gain. A sample scenario is that if m packets correspond to a compressed (e.g., zip) file, the destination node can unzip it only if the entire file of m packets is received. Note that the transport layer end-to-end ACK may not be available for each packet. Even if such ACK is available, it may increase the delay and overhead in data transmission. Therefore, we do not require any packet-level end-to-end ACK. This no packet-level ACK setting was also considered in [13]. Instead, m packets in a file are transmitted for $k \geq m$ times, where k corresponds to a delay bound for this file. We assume there is an ACK for the file after k transmissions (a.k.a. per-group ACK) showing the number of lost packets such that a source node can calculate the past end-to-end packet loss probability and use it as future loss probability. Using this probability, transmission strategy is adapted to improve throughput. Note that there may be estimation error on the end-to-end packet loss probability. The schemes to be designed in Sections III and IV can work with non-optimal parameters but the achieved throughput improvement is smaller than analyzed optimal value.

We consider different packet retransmission and coding schemes applied at the source node to increase the throughput at a destination node. We focus on a single source-destination pair, while the results readily extend to arbitrary source-destination configurations by changing the end-to-end loss probabilities based on the increasing interference effects among multiple sessions.

III. Retransmission Scheme and Performance Analysis

A. Retransmission Scheme

To ensure receiving all packets at the destination node, a source node can employ a ReTransmission (RT) scheme, which can also be regarded as a repetition code. Since there is no end-to-end (per packet) ACK, a source node does not know which packets are lost. Thus, after transmitting each of m packets once, a source node has to retransmit all packets. Moreover, each packet should be transmitted for equal times, whenever possible, since any lost packet may have caused a failure in reception of all m packets. Then, a source node can simply retransmit all packets in the original order until reaching k transmissions. The following is the algorithm for the RT scheme.

Algorithm 1 ReTransmission (RT) Scheme

```
1) For given $m$ and $k$, determine the largest integer $t$ such that $k = tm + c$, where $c$ is a non-negative integer.
2) Transmit $t$ rounds. Within each round, each of the $m$ packets is transmitted once.
3) If $c > 0$, transmit each of the first $c$ packets once.
```

Note that we can also randomly transmit c packets in Step 3 but doing so will not change the expected performance. The simplest way to see this is renaming selected packets as packets

$1, 2, \cdots, c$. Clearly, renaming will not change throughput performance. With new packet names, the random scheme reduces to the deterministic scheme. We adopt the deterministic approach to facilitate performance analysis. Based on the above algorithm, the i-th packet ($1 \le i \le m$) is transmitted in the i-th, $(m+i)$-th, …, $(tm+i)$-th transmissions, if $i \le c$, and is transmitted in the i-th, $(m+i)$-th, …, $((t-1)m+i)$-th transmissions, if $i > c$. Denote

$$K_i = \begin{cases} \{i, m+i, \cdots, tm+i\} & \text{if } i \le c, \\ \{i, m+i, \cdots, (t-1)m+i\} & \text{if } i > c, \end{cases}$$

as the set of transmissions for the i-th packet, and $S_i \subseteq K_i$ as the set of successful transmissions for the i-th packet. If $|S_i| \ge 1$ for each packet i, packet transmissions are successful with throughput $\frac{m}{k}$ (measured as the number of packets received per end-to-end transmission); otherwise, transmissions fail with zero throughput. The i-th packet is received with a probability $Pr(|S_i| \ge 1)$. Under independent loss events in consecutive transmissions (due to changes in channels, topology, and routing or network coding operations), the end-to-end success probability is given by

$$\prod_{i=1}^{m} Pr(|S_i| \ge 1) = \prod_{i=1}^{m} \left(1 - \prod_{j \in K_i} \epsilon_j\right),$$

where ϵ_j is the end-to-end packet loss probability for the j-th transmission. The expected throughput is

$$\lambda_{RT} = \frac{m}{k} \prod_{i=1}^{m} Pr(|S_i| \ge 1) = \frac{m}{k} \prod_{i=1}^{m} \left(1 - \prod_{j \in K_i} \epsilon_j\right). \quad (1)$$

Based on (1), if a source node knows all ϵ_i values, it may identify the optimal value of k for given (fixed) m or the optimal value of m for given (fixed) k to maximize throughput λ_{RT}. A source can learn the average loss probabilities from the destination feedback that needs to be sent less frequently than per-packet acknowledgements. However, the cost to obtain the exact values for all ϵ_i could be still demanding in real-time applications. Instead, we may use the expected end-to-end packet loss probability $\epsilon = E[\epsilon_i]$ over all transmissions to estimate the throughput. Then, we obtain

$$\prod_{i=1}^{m} Pr(|S_i| \ge 1) \approx \prod_{i=1}^{c} (1 - \epsilon^{t+1}) \prod_{i=c+1}^{m} (1 - \epsilon^{t}) = (1 - \epsilon^{t+1})^c (1 - \epsilon^t)^{m-c}$$

and the expected throughput follows from (1) as

$$\lambda_{RT} \approx \frac{m}{k} (1 - \epsilon^{t+1})^c (1 - \epsilon^t)^{m-c}. \quad (2)$$

B. Performance Analysis

We now analyze the RT scheme and choose the best k under given m and ϵ. Note that we can also choose the best m under given k and ϵ. The analysis is similar and thus is omitted.

We first quantify the values of k such that $\lambda_{RT}(k)$ can be a local maximum. We have the following result. The proof is given in the Appendix.

Lemma 1: If $\lambda_{RT}(k)$ is a local maximum, then $k = tm$ for some integer $t \ge 1$.

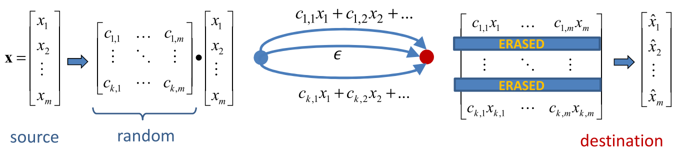


Fig. 1. An illustration of ACO scheme over a lossy network ($\mathbf{x}$ is the vector of all original packets, $\{c_{i,j}\}$ are coding coefficients, and $\hat{x}_i$ is the ith coded packet received at the destination node).

Note that for given m and ϵ, the expected throughput may have multiple local optimal k values, e.g., for $m = 2$ and $\epsilon = 0.45$, $\lambda_{RT}(k)$ is a local maximum at $k = 2$ and $k = 4$.

We then determine the global maximum point for $\lambda_{RT}(k)$. Based on Lemma 1, we focus on considering these values to develop a condition for global optimality. We have the following result. The proof is given in the Appendix.

Theorem 1: For given m and ϵ, the global optimal k is $k = tm$, where t is the smallest integer value such that

$$\frac{t+1}{t} \left(\frac{1 - \epsilon^t}{1 - \epsilon^{t+1}} \right)^m < 1. \quad (3)$$

We note that the optimal value is unique. In general, ϵ_i values may not be the same. We can apply Theorem 1 if ϵ_i values are all approximately equal to a common value ϵ.

IV. Adaptive Packet Coding Optimization and Performance Analysis

A. Adaptive Packet Coding Optimization

In the RT scheme, a source node blindly retransmits all packets. Thus, some already received packets may be retransmitted wasting new transmission opportunities. To resolve this issue, we implement an adaptive packet coding optimization (ACO) to transmit k coded packets (see Fig. 1) as follows.

Algorithm 2 ACO Scheme

```
1) For given $m$ (or $k$), determine a suitable $k$ (or $m$) and
   linearly code $m$ original packets into $k$ packets (to maximize throughput).
2) Transmit each of the $k$ coded packets only once.
```

Denote $K = \{1, 2, \cdots, k\}$ as the set of all coded packets and $S \subseteq K$ as the set of coded packets that are received at the destination. For ease of exposition, we assume that k coded packets are independent and the source-destination pair agrees on the network code. We will consider the case that packets are coded randomly and thus may not be independent, in simulation study. If $|S| \ge m$, the original m packets can be decoded from $|S|$ received packets, which are independent coded packets; otherwise, some packets cannot be decoded and thus the transmission for the group of m packets fails. Such a scheme can also be regarded as an optimized fountain code. For ACO, the success probability is given by

$$Pr(|S| \ge m) = \sum_{S \subseteq K}^{|S| \ge m} \prod_{i \in S} (1 - \epsilon_i) \prod_{i \in K}^{i \notin S} \epsilon_i.$$

We assume that the coding coefficients are transmitted in the packet header along with data and this overhead of m symbols is negligible compared to the total size of data content in each packet. Thus, the expected throughput (the number of packets received per end-to-end transmission) is

$$\lambda_{ACO} = \frac{m}{k} Pr(|S| \geq m) = \frac{m}{k} \sum_{S \subseteq K}^{|S| \geq m} \prod_{i \in S} (1 - \epsilon_i) \prod_{i \in K}^{i \notin S} \epsilon_i. \quad (4)$$

Based on (4), if a source node knows all ϵ_i values, it may identify the optimal k for given m or the optimal m for given k to maximize the throughput λ_{ACO}. Otherwise, a source node needs to estimate the end-to-end loss probabilities as in the RT scheme. In practice, we may use the expected end-to-end packet loss probability ϵ over all transmissions to estimate the achieved throughput. The success probability is given by

$$Pr(|S| \geq m) \approx \sum_{S \subseteq K}^{|S| \geq m} \prod_{i \in S} (1 - \epsilon) \prod_{i \in K}^{i \notin S} \epsilon = \sum_{i=m}^{k} \binom{k}{i} (1 - \epsilon)^i \epsilon^{k-i}$$

and the expected throughput follows from (4) as

$$\lambda_{ACO} \approx \frac{m}{k} \sum\nolimits_{i=m}^{k} \binom{k}{i} (1 - \epsilon)^i \epsilon^{k-i}. \quad (5)$$

Based on this expected throughput, we can maximize the throughput by either choosing a suitable k for given m or choosing a suitable m for given k.

In the ACO scheme, every received packet contributes to the successful reception of the entire file. That is, since each packet appears in different coded combinations, the ACO scheme does not waste any transmission compared to the RT scheme and can achieve better performance. Each original packet undergoes different packet loss events in different coded packet transmissions. This effect generates virtual multipath diversity such that the ACO scheme can achieve better performance than the RT scheme. To show the advantage of ACO, we calculate the expected throughput for $m = 2$, $k = 4$, and $\epsilon = 0.5$. By (2), the expected throughput by the RT scheme is $\lambda_{RT} = \frac{2}{4}(1 - 0.5^2)^2 = 0.28125$. By (5), the expected throughput by the ACO scheme is $\lambda_{ACO} = \frac{2}{4} \sum_{i=2}^{4} \binom{4}{i} (1 - 0.5)^i 0.5^{4-i} = 0.34375$. Thus, ACO scheme can provide 22% throughput gain by coding packets at the source node only. The implementation of the ACO scheme does not require any knowledge on network operations at relay nodes, routing paths, link or node failure probabilities or co-existence of any other source-destination pairs.

B. Performance Analysis

We now analyze the ACO scheme and choose the best k under given m and ϵ. The analysis for choosing the best m under given k and ϵ is similar and thus is omitted. We have the following result. The proof is given in the Appendix.

Theorem 2: For given m and ϵ, denote $\hat{k}$ as the solution to

$$\sum_{i=m}^{k} \binom{k}{i} (1 - \epsilon)^i \epsilon^{k-i} - k \binom{k}{m-1} (1 - \epsilon)^m \epsilon^{k-m+1} = 0. \quad (6)$$

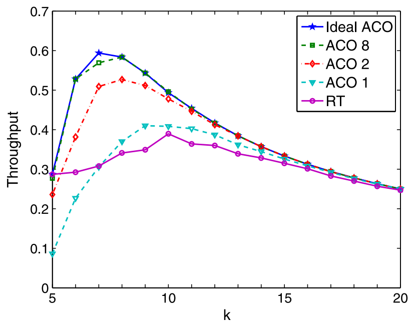


Fig. 2. The throughput achieved by different k when $m = 5$.

Then the best k is either $\lfloor \hat{k} \rfloor$ or $\lceil \hat{k} \rceil$, based on their corresponding λ_{ACO} values.

Note that the optimal k for the ACO scheme is unique unless $\lfloor \hat{k} \rfloor$ and $\lceil \hat{k} \rceil$ achieve the same λ_{ACO} value. In general, ϵ_i values may not be the same. We can apply Theorem 2 if ϵ_i values are all approximately equal to a common value ϵ.

V. Performance Evaluation

We evaluate the throughput performance of ACO and RT schemes through simulation and emulation. In simulation, packets are dropped randomly based on the assumed link loss probabilities, while in emulation packets are lost due to physical channel errors in actual radio transmissions.

A. Simulation Results

We consider a source-destination pair that may choose a different path for each packet. The path length h (the number of hops) is assumed as a random number uniformly distributed in [1, 50]. For simplicity, we assume that the loss probability is 0.01 for all links and the packet loss events at any link are independent from other links. Then the end-to-end packet loss probability is $1 - (1 - 0.01)^h$. Note that in emulation, we will study the case that packets are lost due to errors in actual radio transmissions over emulated wireless channels.

We consider three scenarios:

Scenario 1: A single file is already divided into m packets. Given this m, the source node varies k (the number of coded packets) to find the best k and maximize the throughput. We show the results for $m = 5$ in Fig. 2. The ideal ACO scheme is the one we discussed in Section IV, where the original m packets can be decoded if the number of received coded packets is no less than m. On the other hand, distributed network coding can be achieved via random network codes [14]. Here, the ACO 1, ACO 2, and ACO 8 schemes use random linear codes with field size 2^1, 2^2, and 2^8, respectively. The throughput grows with the field size and approaches the value achieved in the ideal ACO scheme.

Next, we compare the RT scheme and the ideal ACO scheme. When $k = m = 5$, both schemes are successful only if all transmissions are successfully received. Thus, the RT scheme can achieve the same performance as the ideal ACO scheme. When we increase k, both schemes first achieve better performance

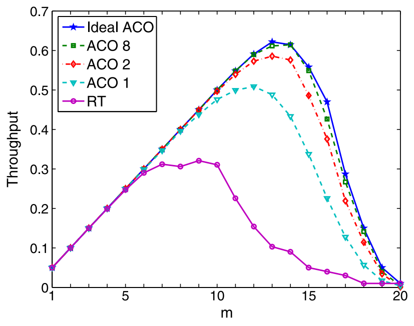


Fig. 3. The throughput achieved by different m when $k = 20$.

because the success probability to receive and decode m packets is increasing. But after a certain point, the throughput starts dropping because the success probability is close to 1 and thus the throughput is close to $\frac{m}{k} = \frac{5}{k}$, which is a decreasing function of k, e.g., for $k = 20$ both schemes achieve the throughput equal to $\frac{5}{20}$.

Based on the results in Fig. 2, the ideal ACO scheme always achieves larger throughput than the RT scheme. As we discussed in Section IV, every transmission under the ideal ACO scheme contributes to the final success, while some transmissions under the RT scheme may be wasted due to retransmissions of packets that are already received. Moreover, the optimal value of k for the ideal ACO scheme is 7 and the achieved throughput is 0.59, while the optimal value of k for the RT scheme is 10 and the achieved throughput is 0.39. In other words, if each scheme chooses the best value of k, the ideal ACO scheme can achieve better throughput performance with less delay.

Scenario 2: There is a delay bound of k end-to-end transmissions for a single file. Given this k, a source node determines the optimal size of packet groups, i.e., the number of packets m, to maximize the throughput. We show the results for $k = 20$ in Fig. 3. We again find that the ACO 8 scheme has almost the same performance as the ideal ACO scheme.

Next, we compare the RT scheme and the ideal ACO scheme. When $m = k = 20$, both schemes again have the same throughput performance. When we decrease m, both schemes first achieve better performance because the success probability to receive and decode m packets is increasing. But after a certain point, the throughput starts dropping because the success probability is close to 1 and the throughput is close to $\frac{m}{k} = \frac{m}{20}$, which is an increasing function of m, e.g., for $m \leq 5$ both schemes achieve a throughput close to $\frac{m}{20}$.

Fig. 3 also shows that the ideal ACO scheme always achieves larger throughput than the RT scheme. Moreover, the optimal m for the ideal ACO scheme is 13 and the achieved throughput is 0.62, while the optimal m for the RT scheme is 9 and the achieved throughput is 0.32. In other words, if each scheme chooses its best value of m, the ideal ACO scheme can achieve better throughput with a different m value.

Scenario 3: The source continually transmits files with different size (i.e., different m), each over different number of transmissions k. We consider total 10^4 packets. Each packet transmission either succeeds or fails, with a successful probability of 0.7.

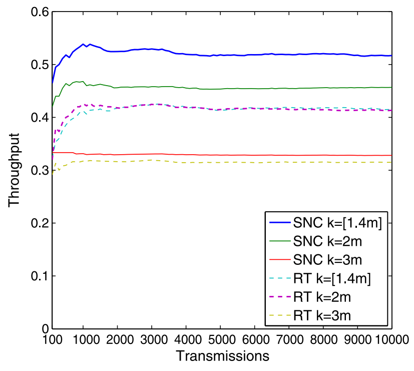


Fig. 4. The throughput achieved by selecting k values.

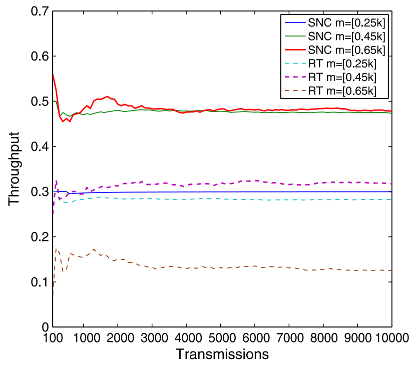


Fig. 5. The throughput achieved by selecting m values.

In what follows, we study two subcases.

Scenario 3.1: The source node has series of files with different m values from [2, 10] and selects k for each m. Although the optimal value of k may be determined (e.g., by taking into account the packet loss probability for each transmission), we simply use fixed k values to demonstrate the advantage of the ACO scheme over the RT scheme. Based on Fig. 2, the ideal ACO scheme has the optimal value close to $k = 1.4m$ and the RT scheme has the optimal value close to $k = 2m$. Thus, we consider $k = [1.4m]$ (where $[x]$ rounds x to the nearest integer), $k = 2m$, and additional case of $k = 3m$ for both RT and ACO schemes.

We compute the average throughput achieved after every 100 transmissions and show it in Fig. 4. With the same k, the ACO scheme always achieves larger throughput than the RT scheme. For the ACO scheme the choice of $k = [1.4m]$ yields better performance than $k = 2m$ or $k = 3m$, while for the RT scheme the choice of $k = 2m$ yields better performance than $k = [1.4m]$ or $k = 3m$. Finally, the ACO scheme with$k = [1.4m]$ sustains the throughput of 0.52, while the RT scheme with $k = 2m$ can only achieve the throughput of 0.41.

Scenario 3.2: The source node has series of files with different k values uniformly drawn from the interval [10, 20] and selects m for each k. Based on Fig. 3, the ideal ACO scheme has the optimal value close to $m = [0.65k]$ and the RT scheme has the optimal value close to $m = [0.45k]$. Thus, we consider $m = [0.65k]$, $m = [0.45k]$, and an additional case of $m = [0.25k]$ for both the RT and ACO schemes. Fig. 5 shows the throughput results. The ACO scheme also achieves larger

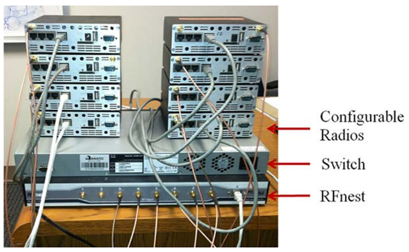
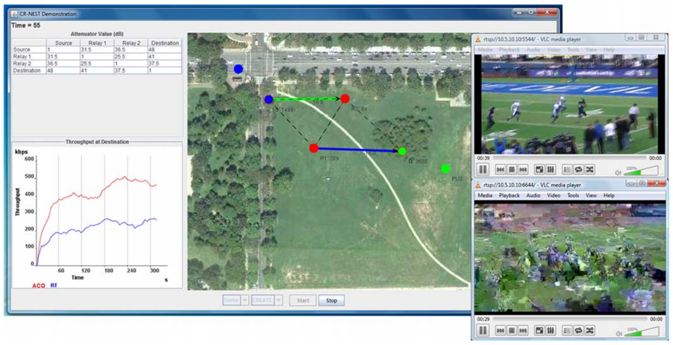


Fig. 6. Programmable RFnest testbed.



Attenuator Value (dB)

	Source	Relay 1	Relay 2	Destination
Source	1	31.5	36.5	48
Relay 1	31.5	1	25.5	41
Relay 2	36.5	25.5	1	37.5
Destination	48	41	37.5	1

Fig. 7. Emulation test GUI.

throughput than the RT scheme for the same choice of k. For the ACO scheme, the choice of $m = [0.65k]$ yields better performance than the other k values while for the RT scheme the choice of $m = [0.45k]$ yields better performance than the other k values. Finally, the ACO scheme with $m = [0.65k]$ sustains the throughput of 0.48, while the RT scheme with $m = [0.45k]$ can only achieve the throughput of 0.32.

B. Emulation Tests

We further evaluated the ACO and RT schemes with our high-fidelity wireless network emulation testbed (as shown in Fig. 6), where wireless radios send real signals over emulated channels and channel attenuation between radios is digitally controlled. There are four main components in our testbed platform: radio frequency network emulator simulator tool, RFnest [12], software simulator running higher-layer protocols on a PC host, configurable RF front-ends (RouterStation Pro from Ubiquiti), and digital switch. Hardware tests are executed at 2.462 GHz channel with 10 dBm transmission power, 10 MHz bandwidth, and 1 Mbps rate. In emulation test GUI (see Fig. 7), we can move nodes and control the network topology. Instead of a sequence of events simulated by some random process (as we did in Section V-A), we record in these emulations the end-to-end packet delivery/loss events generated with real radios.

Transmissions are performed from a source node to a destination node. The source node continually transmits files of m packets (m value may change for each file) over k transmissions, where k is determined by ACO or RT scheme, respectively. Network emulator generates a sequence of events to indicate whether each end-to-end packet delivery over the network is successful or not. The i-th transmission is successful if and only if the i-th event indicates a successful end-to-end packet delivery.

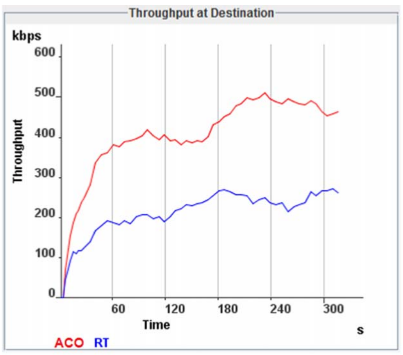


Fig. 8. The throughput achieved in emulation tests.

For emulation results here, we implement a joint channel allocation and routing scheme in a cognitive radio environment [15] over a randomly deployed four-node network, where one node is the source, one node is the destination, and the other two nodes are relays. There are also two primary users, each with one assigned channel. If a primary user is active on its assigned channel, secondary users close to this primary user cannot access this channel at the same time. Both ACO and RT schemes apply to any lossy multi-hop wireless network. As before, the ACO and RT schemes are run at the source and they are oblivious to network operations inside the network. There is no end-to-end per packet ACK. To optimally choose k, we assume there is an end-to-end ACK after a group of k transmissions showing the number of received packets and whether the group of m packets is received. This information is used to estimate the channel condition. Then ACO or RT scheme determines its optimal k for a given m by exhaustive search under the estimated channel condition. As shown in Fig. 2, we observe a single optimal value for k. This can reduce the complexity of search.

Fig. 8 shows the moving average of throughput achieved over time in the emulation tests. Initially, the throughput is zero for both schemes due to the end-to-end propagation delay from the source to the destination. Then, the throughput starts increasing, as more packets are arriving at the destination node. Finally, the throughput converges to some value for each scheme. The emulation results support the analytical and simulation findings and verify that the ACO scheme always achieves better throughput than the RT scheme. To visually show the improvement by ACO via an application, we also run real video traffic in the emulation environment and observed that the video quality significantly improves with the ACO scheme under diverse channel events (see Fig. 7).

VI. Conclusion

In this paper, we implemented an adaptive packet coding optimization scheme for reliable transmissions in a lossy multi-hop network. We studied a scenario that each source node transmits a file of packets to its destination node without per-packet end-to-end ACK because such ACK is not available or causes large delay in a multi-hop network. The coding scheme allows the source node to adaptively code the original packets to maximize throughput, and transmit coded packets to the destination node. Once the destination node receives enough packets, it can decode and obtain all original packets. This scheme does not require any prior knowledge on network topology and routing/network coding protocol inside the network, and adapts to network dynamics measured in terms of end-to-end estimates of average loss probabilities. We showed that by fully exploiting all successful transmissions, this coding scheme can achieve larger throughput than the simple retransmission scheme for unicast traffic. We analyzed its performance and verified the throughput gains via hardware-in-the-loop emulation tests with actual radios.

APPENDIX PROOFS

Proof of Lemma 1: We consider the performance change when we increase k to $k+1$. Recall that we have $k = tm + c$ for some integers t and c. We consider two cases: $c < m-1$ or $c = m-1$. If $c < m-1$, then by (2), the expected throughput when k is increased to $k+1$ is

$$\lambda_{RT}(k+1) = \frac{m}{k+1}(1-\epsilon^{t+1})^{c+1}(1-\epsilon^t)^{m-c+1}.$$

The ratio $r(k)$ between two expected throughputs under k and $k+1$ is

$$\begin{aligned} r(k) &= \frac{\lambda_{RT}(k)}{\lambda_{RT}(k+1)} \\ &= \frac{\frac{m}{k}\prod_{i=1}^m Pr\left(\left|S_i^k\right| \geq 1\right)}{\frac{m}{k+1}\prod_{i=1}^m Pr\left(\left|S_i^{k+1}\right| \geq 1\right)} \\ &= \frac{\frac{m}{k}(1-\epsilon^{t+1})^c(1-\epsilon^t)^{m-c}}{\frac{m}{k+1}(1-\epsilon^{t+1})^{c+1}(1-\epsilon^t)^{m-c+1}} \\ &= \frac{k+1}{k}\frac{1-\epsilon^t}{1-\epsilon^{t+1}}. \end{aligned} \quad (7)$$

If $c = m-1$, the expected throughput when k is increased to $k+1$ is

$$\begin{aligned} \lambda_{RT}(k+1) &= \frac{m}{k+1}\prod_{i=1}^m Pr\left(\left|S_i^{k+1}\right| \geq 1\right) \\ &= \frac{m}{k+1}(1-\epsilon^{t+1})^0(1-\epsilon^t)^m = \frac{m}{k+1}(1-\epsilon^t)^m. \end{aligned}$$

We can verify that (7) also holds.

Next, we analyze the ratio in (7). We need to compare this ratio with 1 to check whether the expected throughput can be increased if we increase k to $k+1$. We first determine how this ratio changes when k is increasing. We again consider two cases: $c < m-1$ or $c = m-1$. It is easy to see that when $c < m-1$, we have

$$\begin{aligned} r(k+1) &= \frac{\lambda_{RT}(k+1)}{\lambda_{RT}(k+2)} \\ &= \frac{\frac{m}{k+1}\prod_{i=1}^m Pr\left(\left|S_i^{k+1}\right| \geq 1\right)}{\frac{m}{k+2}\prod_{i=1}^m Pr\left(\left|S_i^{k+2}\right| \geq 1\right)} \\ &= \frac{k+2}{k+1}\frac{1-\epsilon^t}{1-\epsilon^{t+1}} < \frac{k+1}{k}\frac{1-\epsilon^t}{1-\epsilon^{t+1}} = r(k). \end{aligned}$$

That is, when $c < m-1$, this ratio is decreasing when k is increasing.

When $c = m-1$, we have

$$\begin{aligned} r(k+1) &= \frac{\lambda_{RT}(k+1)}{\lambda_{RT}(k+2)} \\ &= \frac{\frac{m}{k+1}\prod_{i=1}^m Pr\left(\left|S_i^{k+1}\right| \geq 1\right)}{\frac{m}{k+2}\prod_{i=1}^m Pr\left(\left|S_i^{k+2}\right| \geq 1\right)} = \frac{k+2}{k+1}\frac{1-\epsilon^{t+1}}{1-\epsilon^{t+2}}. \end{aligned}$$

In the following, we will show that the difference between two ratios $r(k) = \frac{k+1}{k}\frac{1-\epsilon^t}{1-\epsilon^{t+1}}$ and $r(k+1) = \frac{k+2}{k+1}\frac{1-\epsilon^{t+1}}{1-\epsilon^{t+2}}$ may be positive, zero, or negative when $k = tm + m - 1$. The difference between two ratios is

$$\begin{aligned} &r(k) - r(k+1) \\ &= \frac{k+1}{k}\frac{1-\epsilon^t}{1-\epsilon^{t+1}} - \frac{k+2}{k+1}\frac{1-\epsilon^{t+1}}{1-\epsilon^{t+2}} \\ &= \frac{k+1}{k}\frac{\sum_{i=0}^{t-1}\epsilon^i}{\sum_{i=0}^{t}\epsilon^i} - \frac{k+2}{k+1}\frac{\sum_{i=0}^{t}\epsilon^i}{\sum_{i=0}^{t+1}\epsilon^i} \\ &= \frac{(k+1)^2\sum_{i=0}^{t-1}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i - k(k+2)\left(\sum_{i=0}^{t}\epsilon^i\right)^2}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &= \frac{(k+1)^2\left[\left(\sum_{i=0}^{t}\epsilon^i\right)^2 + (\epsilon^{t+1}-\epsilon^t)\sum_{i=0}^{t}\epsilon^i - \epsilon^{2t+1}\right]}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &\quad - \frac{\left[(k+1)^2-1\right]\left(\sum_{i=0}^{t}\epsilon^i\right)^2}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &= \frac{\left(\sum_{i=0}^{t}\epsilon^i\right)^2 + (\epsilon^{t+1}-\epsilon^t)(k+1)^2\sum_{i=0}^{t}\epsilon^i}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &\quad - \frac{\epsilon^{2t+1}(k+1)^2}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &= \frac{(\epsilon^t-\epsilon^{t+1})\sum_{i=0}^{t}\epsilon^i + \epsilon^{2t+1}}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &\quad \cdot\left[\frac{\left(\sum_{i=0}^{t}\epsilon^i\right)^2}{(\epsilon^t-\epsilon^{t+1})\sum_{i=0}^{t}\epsilon^i + \epsilon^{2t+1}} - (k+1)^2\right] \\ &\approx \frac{(\epsilon^t-\epsilon^{t+1})\sum_{i=0}^{t}\epsilon^i + \epsilon^{2t+1}}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} \\ &\quad \cdot\left[\frac{(\sum_{i=0}^{t}\epsilon^i)^2}{(\epsilon^t-\epsilon^{t+1})\sum_{i=0}^{t}\epsilon^i} - (k+1)^2\right] \\ &= \frac{(\epsilon^t-\epsilon^{t+1})\sum_{i=0}^{t}\epsilon^i + \epsilon^{2t+1}}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i}\left[\frac{\sum_{i=0}^{t}\epsilon^{i-t}}{1-\epsilon} - (k+1)^2\right], \end{aligned}$$

where the "$\approx$" holds when n is large enough. Note that $\frac{(\epsilon^t-\epsilon^{t+1})\sum_{i=0}^{t}\epsilon^i+\epsilon^{2t+1}}{k(k+1)\sum_{i=0}^{t}\epsilon^i\sum_{i=0}^{t+1}\epsilon^i} > 0$, $\frac{\sum_{i=0}^{t}\epsilon^{i-t}}{1-\epsilon}$ increases rapidly with t, and $(k+1)^2 = (tm+m)^2$. Thus, the above difference between two ratios is positive when t is large enough. But if n is not large enough, then the above difference can be positive, zero, or negative.

Combining both cases of $c < m-1$ and $c = m-1$, the ratio in (7) may increase only if $k = tm + m - 1$.

We now compare

$$r(k) = \frac{\lambda_{RT}(k)}{\lambda_{RT}(k+1)} = \frac{\frac{m}{k}\prod_{i=1}^{m} Pr\left(\left|S_i^k\right| \geq 1\right)}{\frac{m}{k+1}\prod_{i=1}^{m} Pr\left(\left|S_i^{k+1}\right| \geq 1\right)}$$

with 1. Since this ratio may increase only if $k = tm + m - 1$, it may achieve the local maximum at $k = tm + m$. Thus, the expected throughput may decrease if we increase k from $tm + m$ to $tm + m + 1$. Then if $\lambda_{RT}(k)$ is a local maximum, k must be either m (the first point) or $tm + m$ for some integer $t \geq 1$.

Proof of Theorem 1: To determine the global maximum point for $\lambda_{RT}(k)$, we can focus on considering $k = tm$ for some integer $t \geq 1$ based on Lemma 1. Consider the change when we increase $k = tm$ to $k = (t+1)m$. We have

$$\frac{\lambda_{RT}(tm)}{\lambda_{RT}(tm+m)} = \frac{\frac{m}{tm}\prod_{i=1}^{m} Pr\left(\left|S_i^{tm}\right| \geq 1\right)}{\frac{m}{(t+1)m}\prod_{i=1}^{m} Pr\left(\left|S_i^{(t+1)m}\right| \geq 1\right)}$$
$$= \frac{\frac{1}{t}(1-\epsilon^t)^m}{\frac{1}{t+1}(1-\epsilon^{t+1})^m} = \frac{t+1}{t}\left(\frac{1-\epsilon^t}{1-\epsilon^{t+1}}\right)^m.$$

We have

$$\frac{\partial\left[\frac{1-\epsilon^t}{1-\epsilon^{t+1}}\right]}{\partial\epsilon} = m\left(\frac{1-\epsilon^t}{1-\epsilon^{t+1}}\right)^{m-1} \cdot \frac{-t\epsilon^{t-1}(1-\epsilon^{t+1}) - (1-\epsilon^t) - (t+1)\epsilon^t}{(1-\epsilon^{t+1})^2}$$
$$= m\left(\frac{1-\epsilon^t}{1-\epsilon^{t+1}}\right)^{m-1}\frac{\epsilon^{t-1}\left[-t + (t+1)\epsilon - \epsilon^{t+1}\right]}{(1-\epsilon^{t+1})^2}.$$

When $t = 1$,

$$-t + (t+1)\epsilon - \epsilon^{t+1} = -1 + 2\epsilon - \epsilon^2 = -(1-\epsilon)^2 < 0.$$

When $t > 1$,

$$\begin{aligned}-t + (t+1)\epsilon - \epsilon^{t+1} &= -t(1-\epsilon) + \epsilon(1-\epsilon^t)\\ &= (1-\epsilon)\left(-t + \epsilon\sum_{i=0}^{t-1}\epsilon^i\right)\\ &< (1-\epsilon)(-t + \epsilon t)\\ &= -t(1-\epsilon)^2 < 0.\end{aligned}$$

Thus, we have

$$\frac{\partial\left[\frac{1-\epsilon^t}{1-\epsilon^{t+1}}\right]}{\partial\epsilon} < 0$$

for any $t \geq 0$, i.e., $\frac{t+1}{t}\left(\frac{1-\epsilon^t}{1-\epsilon^{t+1}}\right)^m$ is a decreasing function of ϵ. Therefore, if $k = tm$ is the global optimal, then t is the smallest integer value such that (3) holds.

Proof of Theorem 2: By (5), the expected throughput when we increase k to $k+1$ is

$$\lambda_{ACO}(k+1) = \frac{m}{k+1}\left[Pr\left(|S^k| \geq m\right) + (1-\epsilon)Pr\left(|S^k| = m-1\right)\right],$$

Then the difference between two expected throughputs under k and $k+1$ is

$$\begin{aligned}&\lambda_{ACO}(k) - \lambda_{ACO}(k+1)\\ &= \frac{m}{k}Pr\left(|S^k| \geq m\right) - \frac{m}{k+1}Pr\left(|S^{k+1}| \geq m\right)\\ &= \frac{m}{k}Pr_{ACO}\left(|S^k| \geq m\right)\\ &\quad - \frac{m}{k+1}\left[Pr_{ACO}\left(|S^k| \geq m\right) + (1-\epsilon)Pr_{ACO}(S^k| = m-1)\right]\\ &= \frac{m}{k(k+1)}Pr_{ACO}(|S^k| \geq m)\\ &\quad - \frac{m}{k+1}(1-\epsilon)\cdot Pr_{ACO}(S^k| = m-1)\\ &= \frac{m}{k(k+1)}\sum_{i=m}^{k}\binom{k}{i}(1-\epsilon)^i\epsilon^{k-i}\\ &\quad - \frac{m}{k+1}(1-\epsilon)\cdot\binom{k}{m-1}(1-\epsilon)^{m-1}\epsilon^{k-m+1}\\ &= \frac{m}{k(k+1)}\left[\sum_{i=m}^{k}\binom{k}{i}(1-\epsilon)^i\epsilon^{k-i} - k\binom{k}{m-1}\cdot(1-\epsilon)^m\epsilon^{k-m+1}\right].\end{aligned}$$

We have

$$\begin{aligned}&\frac{\partial\left[\sum_{i=m}^{k}\binom{k}{i}(1-\epsilon)^i\epsilon^{k-i} - k\binom{k}{m-1}(1-\epsilon)^m\epsilon^{k-m+1}\right]}{\partial\epsilon}\\ &= \sum_{i=m}^{k}\binom{k}{i}\left[(k-i)(1-\epsilon)^i\epsilon^{k-i-1} - i(1-\epsilon)^{i-1}\epsilon^{k-i}\right]\\ &\quad - k\binom{k}{m-1}\left[(k-m+1)(1-\epsilon)^m\epsilon^{k-m} - m(1-\epsilon)^{m-1}\epsilon^{k-m+1}\right]\\ &= -\binom{k}{m}m(1-\epsilon)^{m-1}\epsilon^{k-m} - k\binom{k}{m-1}\\ &\quad\cdot\left[(k-m+1)\cdot(1-\epsilon)^m\epsilon^{k-m} - m(1-\epsilon)^{m-1}\epsilon^{k-m+1}\right]\\ &= -\frac{k!}{(m-1)!(k-m)!}(1-\epsilon)^{m-1}\epsilon^{k-m}\\ &\quad - k\frac{k!}{(m-1)!(k-m)!}(1-\epsilon)^m\epsilon^{k-m}\\ &\quad + mk\frac{k!}{(m-1)!(k-m+1)!}(1-\epsilon)^{m-1}\epsilon^{k-m+1}\\ &= \frac{k!}{(m-1)!(k-m+1)!}(1-\epsilon)^{m-1}\epsilon^{k-m}\\ &\quad\cdot\left[-(k-m+1) - k(k-m+1)(1-\epsilon) + mk\epsilon\right]\\ &= \frac{(k+1)!}{(m-1)!(k-m+1)!}(1-\epsilon)^{m-1}\epsilon^{k-m}\left[k\epsilon - (k-m+1)\right],\end{aligned}$$

where the third equality holds by $\binom{k}{i}(k-i)(1-\epsilon)^i\epsilon^{k-i-1} = \binom{k}{i+1}(i+1)(1-\epsilon)^{(i+1)-1}\epsilon^{k-(i+1)}$ and $\binom{k}{k}(k-k)(1-\epsilon)^k\epsilon^{k-k-1}=0$. By solving

$$\frac{\partial\left[\sum_{i=m}^{k}\binom{k}{i}(1-\epsilon)^i\epsilon^{k-i} - k\binom{k}{m-1}(1-\epsilon)^m\epsilon^{k-m+1}\right]}{\partial\epsilon} = \frac{(k+1)!}{(m-1)!(k-m+1)!}(1-\epsilon)^{m-1}\epsilon^{k-m} \cdot [k\epsilon - (k-m+1)] = 0,$$

we obtain $\epsilon = 0, 1,$ or $\frac{k-m+1}{k}$. Note that $\frac{\partial[\sum_{i=m}^{k}\binom{k}{i}(1-\epsilon)^i\epsilon^{k-i} - k\binom{k}{m-1}(1-\epsilon)^m\epsilon^{k-m+1}]}{\partial\epsilon}$ is negative when $0 < \epsilon < \frac{k-m+1}{k}$ and is positive when $\frac{k-m+1}{k} < \epsilon < 1$. Thus the difference is minimized when $\epsilon = \frac{k-m+1}{k}$. Note that the difference is $\frac{m}{k(k+1)} > 0$ when $\epsilon \to 0$ and approaches zero when $\epsilon \to 1$. Then the minimum difference is negative.

For a given m and a fixed k, Eq. (6) determines a probability threshold $P(m, k)$, which is less than $\frac{k-m+1}{k}$. If $\epsilon < P(m, k)$, then the difference is positive, if $\epsilon = P(m, k)$, then the difference is zero, and if $\epsilon > P(m, k)$, then the difference is negative. Thus for given m and ϵ, if Eq. (6) has an integer solution for k, this solution is the optimal k. Otherwise, denote its solution as $\hat{k}$. The optimal k is either $\lfloor\hat{k}\rfloor$ or $\lceil\hat{k}\rceil$, based on their corresponding λ_{ACO} values.

REFERENCES

[1] 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, Jul. 2000.

[2] M. Medard and A. Sprintson, *Network Coding: Fundamentals and Applications*. London, U.K.: Academic, 2012.

[3] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf Theory*, 49, no. 2, pp. 371–381, Feb. 2003.

[4] Z. Li and B. Li, "Network coding in undirected networks," in *Proc. CISS*, Princeton, NJ, USA, 2004, pp. 257–262.

[5] A. Eryilmaz, A. Ozdaglar, M. Medard, and E. Ahmed, "On the delay and throughput gains of coding in unreliable networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5511–5524, Dec. 2008.

[6] Y. E. Sagduyu and A. Ephremides, "On network coding for stable multicast communication," in *Proc. IEEE MILCOM*, 2007, pp. 1–7.

[7] D. Nguyen and T. Nguyen, "Network coding-based wireless media transmission using POMDP," in *Proc. Packet Video Workshop*, 2009.

[8] S. Katti *et al.*, "XORs in the air: Practical wireless network coding," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 497–510, Jun. 2008.

[9] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and T. Larsen, "Network coding for mobile devices—Systematic binary random rateless codes," in *Proc. IEEE Int Conf. Commun. Workshops*, 2009.

[10] T.-S. Kim *et al.*, "A framework for joint network coding and transmission rate control in wireless networks," in *Proc. IEEE INFOCOM*, 2010.

[11] L. Keller *et al.*, "MicroCast: Cooperative video streaming on smartphones," in *Proc. ACM MobiSys*, 2012, pp. 57–70.

[12] J. Yackoski *et al.*, "RF-NEST: Radio frequency network emulator simulator tool," in *Proc. IEEE MILCOM*, 2011, pp. 1882–1887.

[13] S. H. Y. Wong, R. Raghavendra, Y. Song, and K.-W. Lee, "X-WING: A high-speed wireless broadcasting framework for IEEE 802.11 networks," in *Proc. IEEE SECON*, 2013, pp. 344–352.

[14] T. Ho *et al.*, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[15] L. Ding *et al.*, "CREATE-NEST: A distributed cognitive radio network platform with physical channel awareness," in *Proc. IEEE MILCOM*, 2013, pp. 1669–1674.

Yi Shi (S'02–M'08–SM'13) is a Senior Research Scientist at Intelligent Automation Inc. His current research focuses on algorithms and optimization for social networks, satellite communication system, cognitive radio networks, MIMO, cooperative, sensor and ad hoc networks, and network coding. He has co-organized four IEEE and ACM workshops and has been a TPC member of more than 50 major IEEE and ACM conferences. He is an Editor of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS. He authored one book, five book chapters and more than 100 papers on network design and analysis. He was a recipient of IEEE INFOCOM 2008 Best Paper Award, IEEE INFOCOM 2011 Best Paper Award Runner-Up, and ACM WUWNet 2014 Best Student Paper Award.

Yalin E. Sagduyu (S'02–M'08) received the B.S. degree in electrical and electronics engineering from Bogazici University, Turkey, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland at College Park. He is a Program Manager at Intelligent Automation Inc., Rockville, MD, USA. His research interests are in the areas of design and optimization of wireless networks, cognitive radio networks, network coding, information theory, security, game theory, social networks, and big data analytics. He co-chaired several IEEE and ACM Workshops on cognitive radio networks. He also served as the "MAC and Cross-Layer Design" Track Co-Chair at IEEE PIMRC, 2014.

Junshan Zhang (F'12) is a Professor at the School of ECEE, Arizona State University. His current research focuses on fundamental problems in information networks and energy networks, including modeling and optimization for smart grid, optimization/control of mobile social networks and cognitive radio networks, and privacy/security in information networks. He won the Best Paper Runner-up Award of IEEE INFOCOM 2009 and IEEE INFOCOM 2014, and the IEEE ICC 2008 Best Paper Award. He was TPC co-chair for a number of major conferences in communication networks, including MOBIHOC 2015, INFOCOM 2012, WICON 2008 and IPCCC'06, and TPC vice chair for ICCCN'06. He served as an Associate Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, an editor-at-large for IEEE/ACM TRANSACTIONS ON NETWORKING, and an editor for the *Computer Network* journal, *IEEE Wireless Communication Magazine*, and *IEEE Network Magazine*. He was a Distinguished Lecturer of the IEEE Communications Society.

Jason H. Li (M'02) received the Ph.D. degree in electrical and computer engineering from the University of Maryland, College Park, MD, USA. He is currently the Senior Director of the Networks and Security Group, Intelligent Automation Inc. (IAI), Rockville, MD. Before joining IAI, he was a Researcher with Hughes Network Systems, Germantown, MD. He is the author of more than 40 publications in the area of networks, protocols, security, and multiagent systems. His research interests include computer networks, networks and systems security; cyber security analysis; network management and control; distributed systems; and intelligent software agents. He is a member of the Association for Computing Machinery (ACM), the Advanced Computing Systems Association (USENIX), and the Armed Forces Communications and Electronics Association. He has served on numerous Technical Program Committees for major IEEE/ACM conferences on networks and security-related technologies.