TCP with Hop-Oriented Network Coding in Multi-Radio Multi-Channel Wireless Mesh Networks

Hongquan Liu and Yuantao Gu*

Received Mar. 26, 2012; accepted July 27, 2012.

This article appears in *IET Networks*, 1(3):171-180, 2012.

Abstract

In Multi-Radio Multi-Channel Wireless Mesh Networks (MRMC WMNs), the performance of TCP significantly deteriorates when non-congestion packet losses occur. TCP-HONC is proposed in this paper to improve the performance of TCP with hop-oriented network coding in MRMC WMNs. It combines a block of inter- and intra-flow packets that share a same next-hop node by random linear network coding, and then it forwards and acknowledges these blocks hop by hop. Distributed rate control and path selection algorithm is introduced to effectively utilize network resources and balance load from congested paths to non-congested ones. Simulations show that TCP-HONC achieves a significant throughput gain. Meanwhile, the end-to-end delay and delay jitter of TCP-HONC are relatively small even with a high loss rate. TCP-HONC outperforms TCP/$\mathrm{NC}_{\mathrm{end-to-end}}$ by 236% and TCP/$\mathrm{NC}_{\mathrm{hop-by-hop}}$ by 143% in throughput and reduces the average end-to-end delay to 27% and 90%, when the packet loss rate on each link is 35%. Moreover, resource is fairly allocated to different flows. TCP-HONC is also capable to balance load among different paths and the queue length on bottleneck links is stable. The throughput gain achieved by TCP-HONC is the result of re-encoding data at intermediate nodes and the derived delay reduction benefits from one-hop ACK and retransmission scheme.

Keywords: network coding; wireless mesh networks; load balancing; TCP

1 Introduction

In a typical Wireless Mesh Network (WMN), Internet gateways and wireless mesh routers form a backbone network to provide users with high performance Internet access in a cost-efficient way [1], as illustrated in Fig. 1. In a static single-radio mesh network, all mesh nodes operate on the same channel. As a result, mesh nodes in direct interference range

*This work was partially supported by National Natural Science Foundation of China (NSFC 60872087 and NSFC U0835003). The authors are with Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. The corresponding author of this paper is Yuantao Gu (Email: gyt@tsinghua.edu.cn).

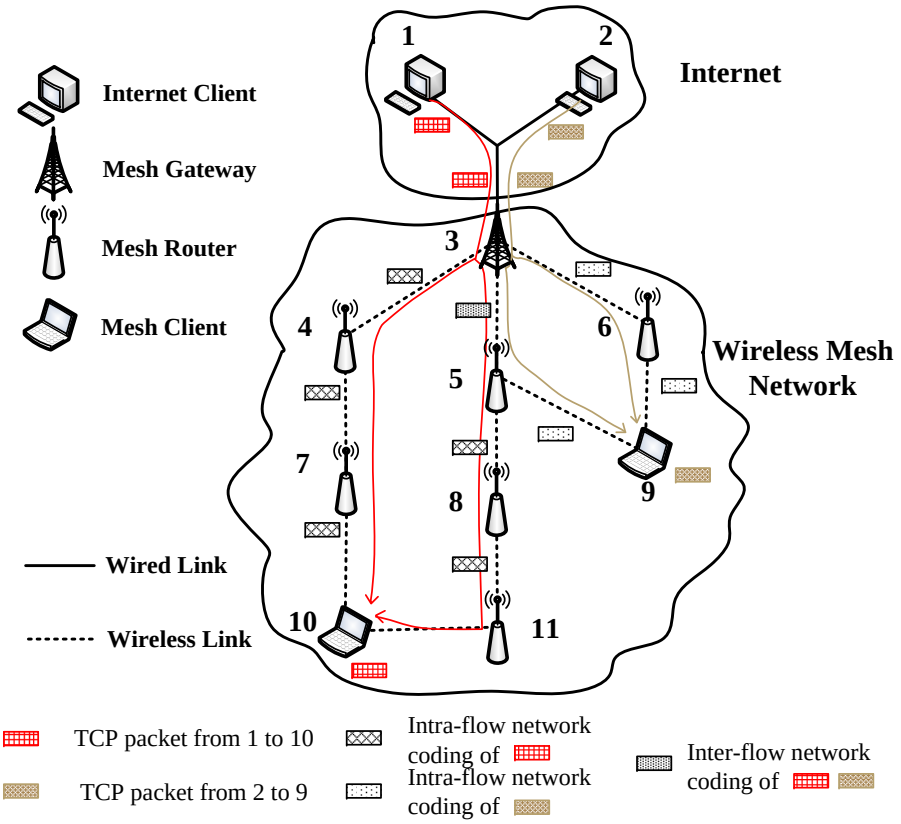

Figure 1: A demonstration of WMNs and an illustration of hop-oriented network coding

of each other contend for the shared channel, so that the available throughput for each node decreases as $O(1/m)$, where m is the number of nodes in the WMN [2]. In Multi-Radio Multi-Channel (MRMC) WMNs, mesh routers are equipped with multiple radios and each radio operates on orthogonal (non-overlapping) channels over a given swath of spectrum. Therefore, nodes can communicate with each other simultaneously in spite of being in direct interference range. Consequently, the available throughput for each mesh node is much better than that in a single-radio mesh network [3].

TCP is the dominant traffic in WMNs nowadays. Therefore, the performance of TCP concerns the popularization and application of WMNs. By reviewing the advances in the research of TCP performance enhancement, Network Coding (NC) has become the state-of-art technique for improving throughput, reducing delay and coping with packet losses [4].

1.1 TCP in WMNs

The throughput of conventional TCP is significantly affected by the peculiarities of multi-hop WMNs. TCP fails to distinguish between congestion and non-congestion losses [5]. Consequently, when both kinds of packet losses happen, a TCP sender reacts in the same way, i.e., it reduces its congestion window by half at least which is unnecessary for non-

congestion losses. As a result, TCP's throughput drops quickly when non-congestion losses happen. Unfortunately, non-congestion losses are very common in WMNs. First, wireless channels suffer from heavy shadow fading which leads to frequently random packet losses. Next, packet collisions and link failures, caused by hidden terminals and mobility in mesh clients, respectively, induce extra non-congestion packet losses. It has been reported that the non-congestion packet loss rate is as high as 50% in IEEE 802.11b based WMNs [6] which typically leads to low TCP throughput.

1.2 Network coding in WMNs

NC allows a wireless node to mix several inter- or intra-flow packets as a single coded packet and then forward it via broadcast. Its neighbors may decode this packet and exact the data that is intended to them only if they have already received enough information. Thus a node may deliver multiple packets in one single transmission. The reduction in the average number of transmission for one packet may translate into a higher throughput. In addition, by doing this, not only is the throughput enhanced but also other performances, such as delay and loss radio in wireless transmission are improved. The improvement of network performance by NC has been demonstrated in theory and implemented in applications (see [4] for an introduction).

1.3 Motivations and contributions

Previous NC approaches have some disadvantages in MRMC WMNs. Firstly most approaches (e.g., COPE [7] and its follow up work [8]) are proposed especially for single-radio mesh networks by using the overhearing opportunity. But this opportunity no longer exists in MRMC WMNs considered in this study because that radio interfaces in direct interference range operate on fixed and orthogonal channels. For example, as seen in Fig. 2, node e communicates with its neighbors via dedicated channels and the radio interfaces on node e don't switch to other channel (or these interfaces is busy all the time). As a result, node e can't get the information about what packets its neighbors have received from other nodes by overhearing. Without overhearing opportunity, these approaches can't work well beyond MRMC. Secondly, the throughput gain derived by NC approaches varies as network topologies, traffic patterns and packet loss rates change. Thirdly, throughput gains achieved by NC may be remarkably lower than expected in practical application [9], in terms of TCP flows, because the TCP congestion control algorithm is not well compatible with the behavior of NC. In addition, merging network coding and TCP together may increases the end-to-end delay. For intra-flow network coding, packets are usually forwarded in block. Let S denote the block size, L and B denote the packet size and bandwidth, respectively. Assuming there are no transmission delay and processing delay, the end-to-end delay is L/B without coding. When using network coding, the end-to-end delay is SL/B at least. Therefore, the combination of NC and TCP may result in an increased end-to-end delay. The delay

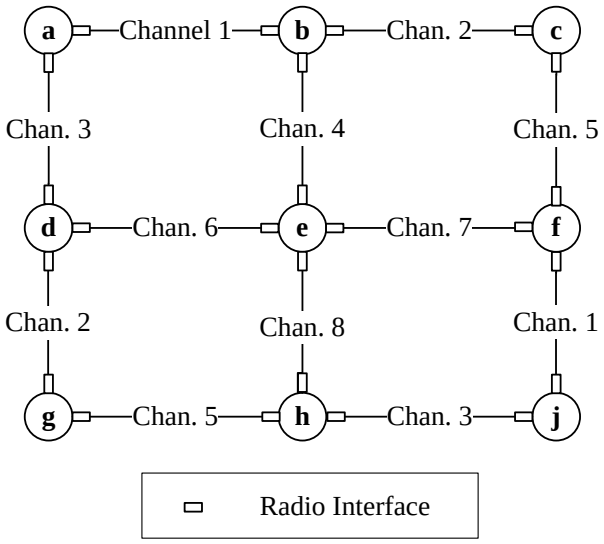

Figure 2: An example of the channel allocation in MRMC WMNs.

performance for NC-based approaches in MRMC WMNs is not well addressed. Finally, most of the previous NC approaches in WMNs don't make use of multipath characteristics for intra-flow data and can't balance load among different paths.

To deal with these issues above, an approach named TCP with Hop-Oriented Network Coding (TCP-HONC) is proposed in this paper. It aims to improve the throughput of TCP flows while maintaining a reasonable end-to-end delay. In addition, it should be capable of providing fairness in resource allocation. The features of TCP-HONC include:

- It is a universal approach in NC applications without the need of overhearing opportunity because it combines inter-flow data that are forwarded to the same direction.
- It presents a way to derive a throughput gain for TCP based on the local information of mesh nodes, no matter what the mesh network architecture, network topology and traffic pattern of TCP flows are.
- It is compatible with TCP. A delay optimization policy is introduced which significantly reduces the end-to-end delay with as little impact as possible on the achieved TCP throughput gain.
- A new path metric is proposed to detect the congestion status of routing paths. With this metric, TCP-HONC is able to balance load from congested paths to non-congested ones.

The remainder of the paper is organized as follows. Section 2 presents related work. Section 3 introduces the network model and basic notions that are used throughout this paper. The proposed approach and the implementation details are presented in section 4

and 5, respectively. Performance evaluation is given in section 6. Finally, section 7 concludes the whole paper.

2 Related work

There is a great deal of follow up work since network coding was firstly introduced in 2000 [10], and the notion of random linear coding [11] facilitates the application of NC.

An intermediate node in COPE [7] collects the information about what packets its neighbors have already received by performing "opportunistic listening", and mixes several packets together only if all the concerned destinations have enough data to extract the information intended to them. Using this method, COPE sends multiple packets in one transmission which obviously increases the throughput in WMNs. However, some experiments by the authors show that the TCP throughput gain by COPE is just 2-3%, due to the high packet loss rate and hidden terminal problems. They also show that COPE derives a throughput gain to 38% without hidden terminals. Seferoglu et al. [8] focus on the performance of COPE in the presence of non-negligible random losses and propose an intra- and inter-session network coding (I^2NC) approach. I^2NC is resilient to loss and can work without the exact information of the state of neighbors. Simulations in GloMoSim demonstrate I^2NC outperforms COPE. However, the throughput gain of COPE and its follow up work is built on specific topologies (i.e., X topology, Cross topology, etc.) and traffic patterns with sufficient load (e.g., bidirectional flows). In addition, both COPE and I^2NC require overhearing opportunity and can't balance load among different paths.

TCP/NC [12–14] is the first approach that merges network coding with TCP. Inspired by TCP/NC, Senger et al. [15] also proposed a similar approach. These two approaches are stream-based. They perform intra-flow network coding and mask packet losses from the TCP congestion control algorithm. In addition, they are compatible with the TCP sliding window. Simulations show that they achieve much higher throughput compared with TCP in lossy wireless networks. But the throughput gains rely heavily on the value of redundancy parameter which is related to the packet loss rate of routing paths. For example, with an inappropriate value of redundancy parameter, the throughput may reduce to 2.716%. Unfortunately, packet loss rate isn't a constant in reality. Therefore, the redundancy can't always be set to an appropriate value. As a result, the achieved throughput is lower than expected in practical application. In addition, the performance of end-to-end delay and delay jitter of TCP/NC weren't studied in [12–14].

TCP/NC is based on end-to-end coding. But the authors had claimed that TCP/NC also allows intermediate nodes to perform re-encoding of data, which can achieve significant benefits in throughput in a single path unicast scenario with multiple lossy hops. However, actually, when intermediate nodes perform re-encoding of data, the achieved throughput is much lower than expected.

Gheorghiu et al. [16] extend TCP/NC to multipath forwarding by exploiting the path

diversity. The rate of transmitted linear combinations is controlled based on a credit-based method. Intermediate nodes are allowed to generate new linear combinations whenever packet loss is detected. Simulations show that this follow-up approach slightly outperforms TCP/NC. Both TCP/NC and its follow-up work are based on the traditional congestion control mechanism.

MPLOT [17–19] was proposed to utilize the available bandwidth and path diversity in lossy WMNs. It employs Forward Error Correction (FEC) to add redundancy to cope with random losses. An intelligent packet mapping module is introduced based on current path characteristics, i.e., packet loss rate, Round-Trip Time (RTT) and bandwidth, which would reduce the number of out-of-order packets. MPLOT is able to effectively exploit path diversity and achieve higher throughput than traditional TCP-SACK even with packet losses as high as 50%. However, it is an end-to-end coding approach and can't effectively utilize available bandwidth in lossy multi-hop wireless networks because it may waste the bandwidth of links where redundancy is not needed.

Joint routing and scheduling in WMNs also have received great attention. Li et al. [20] formulated the joint multi-path routing and MAC scheduling as a utility maximization problem and proposed a solution to this problem which enhances the rates of network resource utilization efficiency. Tang et al. [21] investigated the problem of joint routing and scheduling in WiMAX mesh networks and proposed three sets of algorithms to make efficient use of network resource.

3 Model

In this section, a packet-switched network model developed in [22,23] is extended to MRMC WMNs and basic notions used throughout this paper are introduced.

3.1 Network model

For a lossy MRMC WMN, the network is modeled as a directed $G = (V, E)$, where V is the set of mesh nodes and E is the set of wireless hops. Node m has $\kappa(m)$ radios and each radio has K orthogonal channels, where $\kappa(m) \leq K$. $N(m)$ is the number of neighbors for node m. Assuming that $\kappa(m) \geq N(m)$, it is possible that node m communicates with each neighbor via a dedicated radio and orthogonal channels are allocated to wireless links that are in the interference range, which means that a) the radios on each node don't need to switch between channels, b) all the wireless links are interference-free and c) adjacent links may transmit packets simultaneously. A Time-Division Multiple Access (TDMA) mechanism is adopted and the wireless links are scheduled periodically.

Hop (m, n) denotes a packet-erasure point-to-point link. A packet injected into hop (m, n) is either lost or received by node n without error. Hop (m, n) has three parameters which are c_{mn}, d_{mn} and p_{mn}, where c_{mn} is the channel capacity, d_{mn} is the propagation

delay and p_{mn} is the packet loss rate. For $\tau \geq 0$, $s_{mn}(\tau)$ is the total number of packets sent between time 0 and time τ and $r_{mn}(\tau)$ is the total number of packets received between time 0 and time τ on hop (m, n). Then $p_{mn} = 1 - lim_{\tau \to \infty} s_{mn}(\tau)/r_{mn}(\tau)$. A lossy MRMC WMN is defined as a group (G, c, d, p).

The main characteristics of MRMC WMNs considered in this paper are summarized in what follows.

- Mesh gateways and mesh routers are immobile and they communicate with each other on dedicated radios and orthogonal channels.
- Each radio operates on one dedicated channel which doesn't change with time.
- These wireless links are TDMA and bi-directional.
- There are usually multiple paths from sources to destinations.

3.2 Traffic, routing and flow scheduling

Let $\mathbf{S}$ be the set of unicast TCP flows, and each flow $s \in \mathbf{S}$ has a source node, $Src(s) \in V$ and a destination node, $Dst(s) \in V$, and is associated with a rate f_s.

Each flow s travels along either a single path or two paths from a source to a destination which is calculated in advance by a routing protocol, e.g., LBM [24], LET [25], MR-LQSR [26], MR-AODV [27, 28], and given as input to our problem. For a flow s with two paths, TCP packets are scheduled between these two paths according to our proposed path selection algorithm.

4 Protocol Description

In this section, the proposed protocol TCP-HONC is presented in details. A wireless mesh node in our system is allowed to mix and split TCP packets in a hop-oriented way. On the one hand, each node chooses a block of original packets from different flows which share a same next-hop node, performs NC and then sends coded packets to next hop. On the other hand, each node decodes the received encoded blocks from different links into original packets, and stores them in different buffers according to their flow numbers. Encoding and decoding on each node perform in a manner of Hop-Oriented Network Coding (HONC), which is described in subsection 4.1. The rate control algorithm is introduced in subsection 4.2 to avoid congestions. A path selection algorithm is introduced in subsection 4.3 to intelligently map packets to different paths that given by the routing protocol.

4.1 Hop-oriented network coding

For hop (m, n), assume there are N_{mn} TCP flows that travel from m to n. Let L_{mn} denote the size of the encoding block delivered through hop (m, n), one has

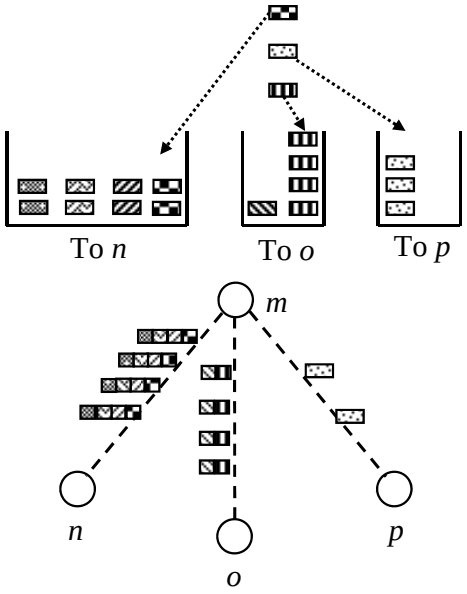

Figure 3: A mesh node in TCP-HONC processes TCP packets

$$L_{mn} = \begin{cases} N_{mn}, & N_{mn} \geq l, \\ l, & N_{mn} < l. \end{cases} \tag{1}$$

Usually, l is empirically set as 4. As illustrated in Fig. 3, mode m first buffers the original packets of these N_{mn} TCP flows separately according to their flow numbers. Then it uniformly picks L_{mn} packets from these N_{mn} buffers and performs random linear network coding [11] by randomly choosing each coding coefficient over a Galois Field $GF(256)$. $\lfloor R_{mn}L_{mn} \rfloor$ combinations are forwarded to node n for each encoding block to cope with random packet losses, where R_{mn} is the redundancy parameter over hop (m, n) and is given by

$$R_{mn} = 1/(1 - p_{mn}). \tag{2}$$

After L_{mn} coded packets of one encoding block with independent coefficients are received, node n decodes this block and buffers these L_{mn} original TCP packets. Next, an ACKnowledgement for Network Coding Block (ACK-NCB) is generated and sent to node m. Node m deletes this block of original packets while receiving this ACK-NCB. If node n fails to decode these packets, node m will receive an ACK-NCB with the information about the rank of coefficient matrix at node n. Next, node m retransmits $\lfloor R_{mn}L_{mn} \rfloor - r$ additional linear combinations of this block, where r denotes the rank of coefficient matrix. To be noticed, there are multiple codec pairs over hop (m, n) in order to effectively utilize network resource. By our HONC scheme, packets are reliably forwarded to their destinations hop by hop.

HONC scheme is only performed on wireless links, TCP packets between mesh gateways and Internet nodes are forwarded in the original way. Figure 1 illustrates how this scheme

works. There are two TCP flows, one travels along the two red paths and the other travels along the two gray paths. On the wired side, nothing is changed and original TCP packets are forwarded. On hop (3, 5), there are two TCP flows, therefore, node 3 performs one-hop inter- and intra-flow network coding. On the other wireless hops, there is only one flow, therefore, the sender of each hop performs one-hop intra-flow network coding. Beyond this hop-by-hop inter- and intra-flow network coding, TCP packets are reliably forwarded from sources to destinations in block.

The benefits of introducing HONC in our system are summarized as follows.

- **Effective to deal with losses**: HONC masks losses from TCP, guarantees reliability and reduces retransmission delay by introducing redundancy and one-hop ACK and retransmission scheme. Redundancy would help a receiver to decode those L_{mn} original packets without retransmissions with a high probability, which can get rid of retransmission delay. Furthermore, when having a decoding failure, it would take less time to send additional coded packets by one-hop ACK and retransmission scheme, compared with end-to-end ACK and retransmission scheme.

- **Easy to fairly allocate network resource**: HONC scheme forwards L_{mn} original packets each time, and it is easy to uniformly choose packets from different flow buffers. Therefore, fair resource allocation is achieved by HONC scheme.

- **Effective to utilize network bandwidth**: HONC scheme allows intermediate nodes to generate new packets, which give us the flexibility to add redundancy for covering random packet losses only before lossy links. On the contrary, an end-to-end coding scheme wound waste the bandwidth of links where redundancy is not needed.

4.1.1 Throughput gain

There is a throughput gain between hop-by-hop coding scheme and end-to-end coding scheme in a single-path unicast scenario with multiple lossy hops.

Assuming that there are n wireless hops and each hop has a same bandwidth b and packet loss rate p, the maximum throughput achieved by these two schemes in theory are given by

$$Thr^n_{hop-by-hop} = b(1-p), \tag{3}$$

$$Thr^n_{end-to-end} = b(1-p)^n. \tag{4}$$

Finally, the throughput gain is obtained by

$$Thr^n_{gain} = \frac{Thr^n_{hop-by-hop}}{Thr^n_{end-to-end}} = (1-p)^{1-n}. \tag{5}$$

4.1.2 Delay reduction

TCP-HONC employs one-hop ACK and retransmission scheme which can cover random losses only before lossy links and obtain a delay reduction, compared with an end-to-end ACK and retransmission scheme.

In the single path unicast scenario of section 4.1.1, assuming that transmission delay can be neglected compared with propagation delay, the end-to-end delay for both one-hop and end-to-end ACK and retransmission schemes are given in (6) and (7), respectively,

$$Delay^n_{one-hop} = \frac{nd(1+p)}{1-p}, \tag{6}$$

$$Delay^n_{end-to-end} = \frac{d(1+p^2)}{p(1-p)^n} - \frac{d}{p} + dp\sum_{i=0}^{n-2}\frac{n-i}{(1-p)^{i+1}}, \tag{7}$$

which are proved in appendix.

Finally, the delay reduction is given by

$$\begin{aligned} Delay^n_{reduction} &= \frac{Delay^n_{one-hop}}{Delay^n_{end-to-end}} \\ &= \frac{np(1+p)(1-p)^{n-1}}{(1+p^2)-(1-p)^n + p^2\sum_{i=0}^{n-2}(n-i)(1-p)^{n-i-1}}, \end{aligned} \tag{8}$$

and $Delay^n_{reduction} = 1 \,\text{when}\, p = 0$.

4.2 Rate control

The congestion control algorithm of conventional TCP isn't able to adapt the wireless environment is the major reason why the TCP throughput decreases sharply as random packet losses increase. A distributed rate control algorithm is introduced based on back-pressure method [29] to greatly minimize the interaction between the conventional TCP and WMNs and make HONC work effectively and smoothly.

Firstly, the available number of codec pairs over hop (m, n), which is a key parameter in TCP-HONC, is adaptively changed. The optimal value of this number should guarantee that there are enough codec pairs to fully utilize capacity resources (i.e., when the channel is idle, there are still free codec pairs that can be used whenever necessary), meanwhile coded packets that cannot be afforded by the network won't be injected into it. In the ideal case (i.e., packet loss rate is zero), this number, denoted as B'_{mn}, is given by

$$B'_{mn} = \left\lceil \frac{2d_{mn} + L_{data}L_{mn}/c_{mn}}{L_{data}L_{mn}/c_{mn}} \right\rceil, \tag{9}$$

where L_{data} is the packet size. For a certain codec pair, packet losses may lead to decoding failure.

As a result, this pair can't be used to forward new TCP packets later. Therefore, the number of available codec pairs, denoted as B_{mn}, needs to be increased to effectively utilize

network resource when losses occur. The probability that node n successfully decodes one block of packets is given by

$$P_{bmn} = \sum_{k=L_{mn}}^{k \leq \lfloor R_{mn} L_{mn} \rfloor} \binom{k}{\lfloor R_{mn} L_{mn} \rfloor} (1 - p_{mn})^k p_{mn}^{\lfloor R_{mn} L_{mn} \rfloor - k}. \tag{10}$$

Let $\mathbf{B}$ be the set of values, with which the probability of no less than B'_{mn} codec pairs having successful decoding during a first round transmission is greater than P_{th}, where P_{th} is a threshold value and is set to 0.8 in our system. Finally, the proper B_{mn} in the proposed system is the minimum value in $\mathbf{B}$, which is given by

$$B_{mn} = \min \arg \left(\sum_{k=B'_{mn}}^{k \leq B^*_{mn}} \binom{k}{B^*_{mn}} p_{mn}^k (1 - p_{mn})^{B^*_{mn} - k} \geq P_{th} \right). \tag{11}$$

On hop (m, n), node m uses a queue Q_{emn} to buffer the original TCP packets that need to be forwarded over this hop. Let Q_{smn} be the number of packets in the sending queue on this hop. Let B''_{mn} denote the number of codec pairs that are currently in use. The proposed distributed rate control algorithm is given as what follows.

Node m forwards coded packets to node n in blocks as long as (12) – (15) are satisfied,

$$\sum_{k \in H_{mn}} (Q_{enk} + Q_{snk}) \leq (B_{mn} - B'_{mn} + 1) L_{mn}, \tag{12}$$

$$Q_{smn} = 0, \tag{13}$$

$$B''_{mn} < B_{mn}, \tag{14}$$

$$Q_{emn} \geq L_{mn}, \tag{15}$$

where H_{mn} is the set of nodes, by which TCP flows over hop (m, n) are forwarded. If m is a mesh client or a mesh gateway and only satisfies (12) – (14), it gets extra packets by spoofing, as illustrated in Fig. 4.

4.3 Path selection

Multipath delivery are introduced in TCP-HONC to exploit the path diversity and effectively utilize network resource. TCP-HONC introduces a path (or next hop) selection module which is deployed on mesh nodes that have multiple paths to destinations to intelligently map packets to different paths. A path estimation algorithm is also introduced to collect the cost of each path for each TCP flow based on the proposed HONC scheme. The detail of this path selection algorithm is described as follows.

Let C^s_m be the estimated minimal cost of path that from node m to the destination of flow s, then C^s_m is calculated as follows.

$$n(s) = \arg \min_{n \in N_s(m)} \left(\frac{|Q_{emn}| + |Q_{smn}|}{c_{mn}(1 - p_{mn})} + C^s_n \right), \tag{16}$$

$$C^s_m = \frac{|Q_{em,n(s)}| + |Q_{sm,n(s)}|}{c_{m,n(s)}(1 - p_{m,n(s)})} + C^s_{n(s)}, \tag{17}$$

where $N_s(m)$ is the set of next-hop nodes from node m to the destination of flow s. For hop (m, n), C_n^s is piggybacked on the ACK-NCBs from node n to node m, and $C_n^s = 0$ if n is the destination of flow s.

When node m receives a packet of flow s, it chooses the next hop given by (16).

5 Protocol implementation

The implementation of all these components mentioned in section 4 needs to be done with minor modifications to the existing protocol stack. Meanwhile it should be compatible with the conventional TCP. TCP-HONC introduces a thin network coding layer below Internet Protocol and above Link Layer on all the wireless nodes. The new layer filters TCP packets from both upstream and downstream data. It takes a number of operations when it sends (Alg. 1) and receives (Alg. 2) a packet.

Alg. 1 Node m sends packets to node n

While(true)

1) If hop (m, n) satisfies (12)–(15), select L_{mn} inter-flow TCP packets from Q_{emn} as a coding block whose index is i, generate and send $\lfloor R_{mn} L_{mn} \rfloor$ random linear combinations of this block, recalculate $s_{mn}, B''_{mn}, Q_{emn}, Q_{smn}$.

2) If Receiving Timer (k, i) expires, generate and send an ACK-NCB to node k with some certain information.

3) If m is the destination or mesh gateways, send the packets in sequence from Q_{em} (Q_{em} is the queue that sequentially buffers decoded TCP packets) to higher layer or Internet-side clients.

Information in an ACK-NCB: The decoder (block) index i, the rank of the decoding coefficient matrix (denoted as r_i), the estimated propagation delay d_{km}, C_m^f and r_{mn} are piggybacked on this packet.

Receiving Timer (k, i): Receiving Timer (k, i) is deployed on node m to avoid the following situation occurs: the decoder (whose index is i) still can't successfully decode this block when the encoder has already sent $\lfloor R_{mn} L_{mn} \rfloor$ packets. The timeout interval is set to $\Delta T_{km} L_{km}$, where ΔT_{km} is the arrival interval of packets from node k.

Alg. 2 Node m receives packet b from node k

1) If b is a TCP control packet, send it directly;
2) else if b is a TCP packet, then select next-hop node n based on (16) and insert b to Q_{emn} if $b \notin Q_{emn}$;
3) else if b is an ACK packet,
   a) if m isn't a destination or mesh gateway, drop it,
   b) else, delete relevant buffered TCP packets in Q_{em};
4) else if b is a NC packet, assuming (m, n) is b's next hop and its block index is i, decode it,
   a) if successfully decoded, send an ACK-NCB to k, save decoded packets in Q_{emn}, cancel the Receiving Timer (k, i),
   b) else, restart the Receiving Timer (k, i).
5) else if b is an ACK-NCB packet whose block index is j,
   a) if $r_j = L_{mk}, B''_{mk} - -$, recalculate p_{mk}, d_{mk}, recalculate $B'_{mk}, B_{mk}\, and\, C^f_m$ according to (9), (11) and (18),
   b) else, re-send $\lfloor R_{mk} L_{mk} \rfloor - r_j$ NC packets of block j

Recalculate p_{mk}, d_{mk}: p_{mk} and d_{mk} are estimated by the way that RTT in TCP does, as shown in (18), (19). p_{mk_this} is equal to $1 - r_{mk}/s_{mk}$, and d_{mk_this} is the d_{mk} value obtained in an ACK-NCB currently.

$$p_{mk\_new} = \alpha p_{mk\_old} + (1 - \alpha) p_{mk\_this}, \tag{18}$$

$$d_{mk\_new} = \alpha d_{mk\_old} + (1 - \alpha) d_{mk\_this}, \tag{19}$$

where the smoothing factor α is set to 0.875.

6 Performance evaluation

In this section, simulation results which are obtained with NS-2 [30] are presented. For the rest of the paper, we refer to the approach proposed in [12,13] as TCP/NC$_{\mathrm{end-to-end}}$, and we extend TCP/NC$_{\mathrm{end-to-end}}$ to a hop-by-hop coding approach, denoted as TCP/NC$_{\mathrm{hop-by-hop}}$, by performing re-encoding of data at intermediate nodes. Network topology used in the simulation is illustrated in Fig. 1. The bandwidth and propagation delay on wired links are 100 Mbps and 1 ms, respectively. The capacity and propagation delay on wireless links are 1 Mbps and 100 ms, respectively. The receive window size of TCP is set to 100 packets. There is no link layer retransmission in all these simulations. Each wireless link has a same packet loss rate, denoted as P_e. The simulation time is 10000s if it isn't specified. These approaches are evaluated in both single path and multi-path scenarios.

6.1 Single path scenario

There are 4 FTP applications that send packets from node 1 to 10 along path of node 1-3-5-8-11-10 simultaneously. The throughput and end-to-end delay performance is first studied.

As shown in Fig. 5 (a), the throughput of TCP decreases exponentially as losses increase. TCP/NC$_{\mathrm{end-to-end}}$ derives a throughput close to $Thr_{end-to-end}$, however, it is worse than TCP/NC$_{\mathrm{hop-by-hop}}$ as losses increase because the latter allows to re-encode data at intermediate nodes. But TCP/NC$_{\mathrm{hop-by-hop}}$ doesn't fully utilize network bandwidth when loss rate is high, compared with TCP-HONC. Our approach obtains a throughput which is close to $Thr_{hop-by-hop}$. It outperforms TCP/NC$_{\mathrm{end-to-end}}$ by 236% and TCP/NC$_{\mathrm{hop-by-hop}}$ by 143% in throughput when the packet loss rate on each wireless link is 35%. TCP-HONC greatly exploits the benefit of re-encoding and obtains a throughput gain close to the theoretical value, as seen in Fig. 5 (d).

As shown in Fig. 5 (b), the end-to-end delay of TCP remains small when the packet loss rate is low, but it has an exponential growth after the loss rate is higher than 5%. The delay performance of both TCP/NC$_{\mathrm{end-to-end}}$ and TCP/NC$_{\mathrm{hop-by-hop}}$ is also not satisfactory. Comparing with them, TCP-HONC obtains a small and slowing-growth delay as losses increase, as illustrated in Fig. 5 (b), and achieves a better delay reduction than expected in most cases, as shown in Fig. 5 (d). This delay reduction benefits from adding redundancy and performing one-hop ACK and retransmission scheme, as explained before.

The standard deviation of end-to-end delay of TCP-HONC is also low, which means TCP-HONC also has a very small delay jitter, as shown in Fig. 5 (c).

Next, the error performance is studied. Redundancy and propagation delay on each hop are two key parameters that greatly affect the performance of TCP-HONC.

The theoretical optimum value of redundancy for TCP/NC$_{\mathrm{end-to-end}}$ is $1/(1-P_e)^4$ and the value for TCP/NC$_{\mathrm{hop-by-hop}}$ and TCP-HONC is $1/(1-P_e)$. Therefore, we may study the effect of estimated loss rate on the throughput instead of redundancy parameter. As shown in Fig. 6, TCP/NC$_{\mathrm{end-to-end}}$ and TCP/NC$_{\mathrm{hop-by-hop}}$ achieve their peak throughput near the actual loss rate, however their throughput decrease quickly as the estimated loss rate goes away from the actual one. The throughput of TCP-HONC is relatively stable as the estimated loss rate changes, which means that TCP-HONC achieves a good performance even with an inaccurate estimation of loss rate, and this means a lot to practical applications.

Propagation delay on each hop is estimated by TCP-HONC to calculate the number of available codecs. As seen in Fig. 7, when the estimated propagation delay is getting lower than the actually one (i.e., 0.1s), the achieved throughput and end-to-end delay are getting a little smaller; when it is getting larger, the achieved throughput remains the same but the end-to-end delay is getting bigger. The reasons can be briefly stated. According to equation (9), the estimated propagation delay is in direct proportion to the number of available codecs B'_{mn}. A small B'_{mn} means there may not be enough codecs to fully utilize

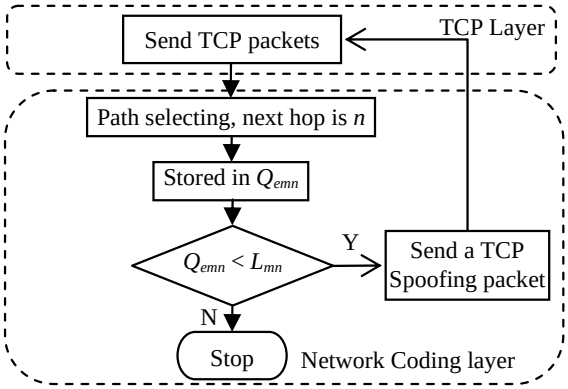

Figure 4: The way source nodes get extra TCP packets by TCP Spoofing

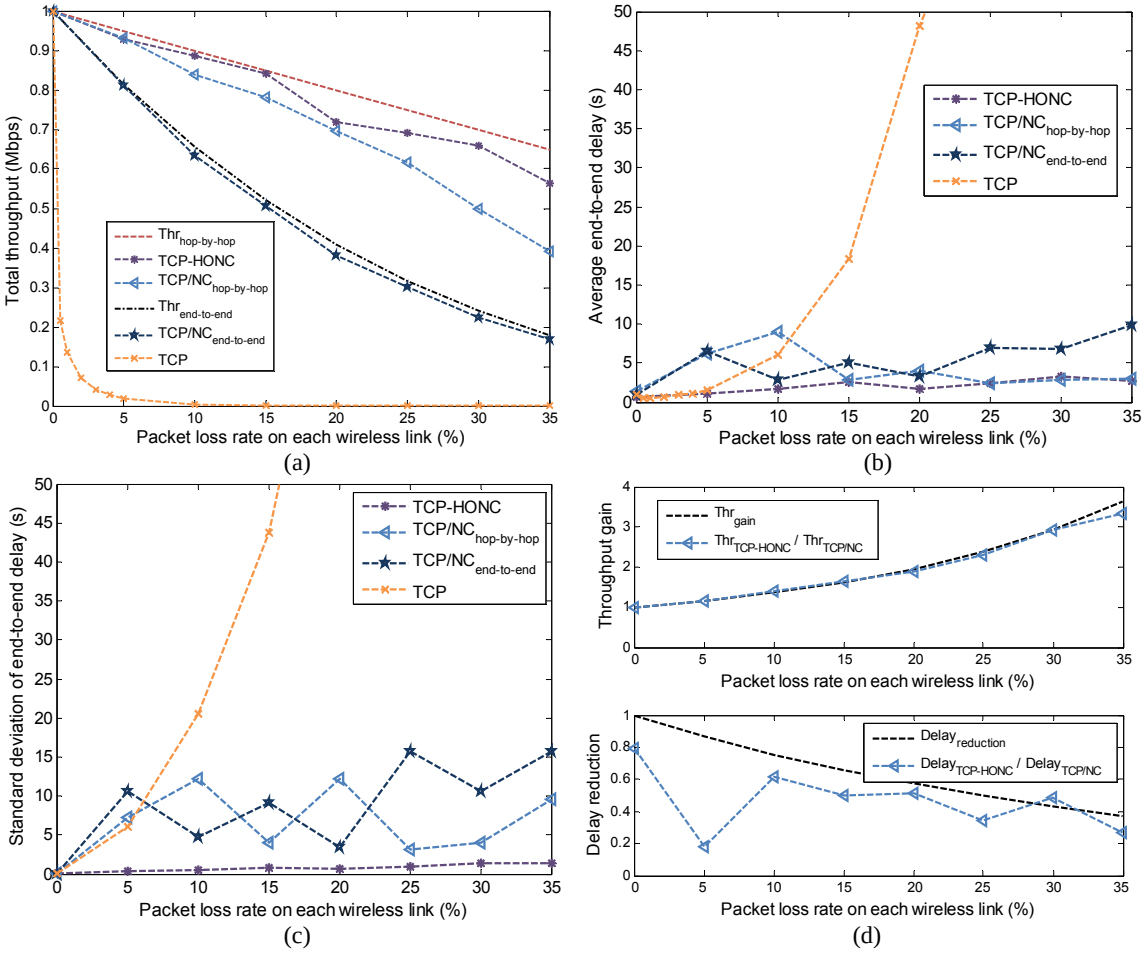

Figure 5: Results of single-path scenario under various packet loss rate: (a) total throughput; (b) average end-to-end delay; (c) standard deviation of delay; (d) throughput gain and delay reduction.

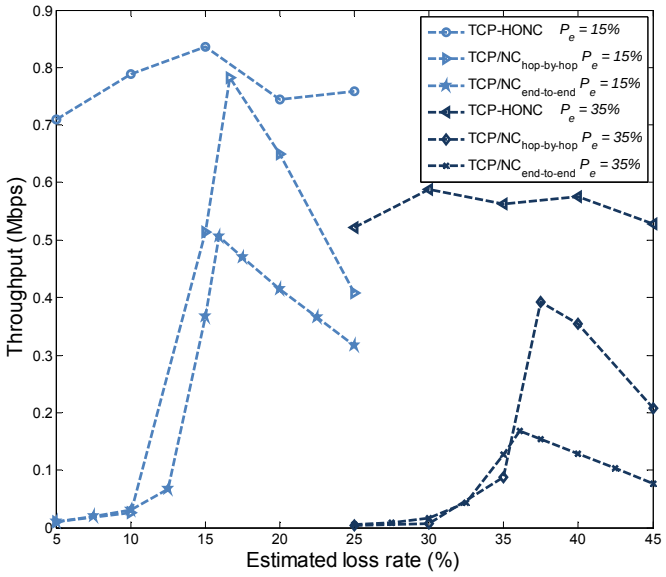

Figure 6: Throughput vs estimated loss rate

the bandwidth. As a result, the achieved throughput may be small. However, a small B'_{mn} also makes the queuing delay is small which brings us a small end-to-end delay. After all, the effects of estimated propagation delay on the throughput and end-to-end delay are relatively small and our approach is very robust in practical application.

6.2 Multi-path scenario

For the multi-path simulation scenario, packets from node 1 to node 10 are forwarded along two asymmetrical paths, i.e., the two red paths in Fig. 1.

6.2.1 Throughput and end-to-end delay

The throughput and end-to-end delay performance is also first studied.

As shown in Fig. 8 (a), TCP-HONC fully utilizes the benefit of two routing paths and derives a total throughput which is close to the total $Thr_{hop-to-hop}$. The distance between TCP/NC$_{\mathrm{end-to-end}}$ and $Thr_{end-to-end}$ is larger than that in Fig. 5 (a), and TCP/NC$_{\mathrm{hop-by-hop}}$ is even worse than TCP/NC$_{\mathrm{end-to-end}}$ in throughput, which means they doesn't effectively use the bandwidth of two routing paths. This is caused by the asymmetry of routing paths. Asymmetrical paths cause many out-of-order packets which may decrease the throughput of TCP/NC. TCP-HONC far outperforms TCP/NC$_{\mathrm{end-to-end}}$ and TCP/NC$_{\mathrm{hop-by-hop}}$ in multiple paths as losses increase. It derives a total throughput which is 347% of the throughput of TCP/NC$_{\mathrm{end-to-end}}$ when the loss rate is 35%. The end-to-end delay of TCP increases quickly as packet loss rate is getting bigger; the delay of TCP-HONC is also relatively small when losses increase, as shown in Fig. 8 (b). But the delay of TCP/NC$_{\mathrm{end-to-end}}$ and

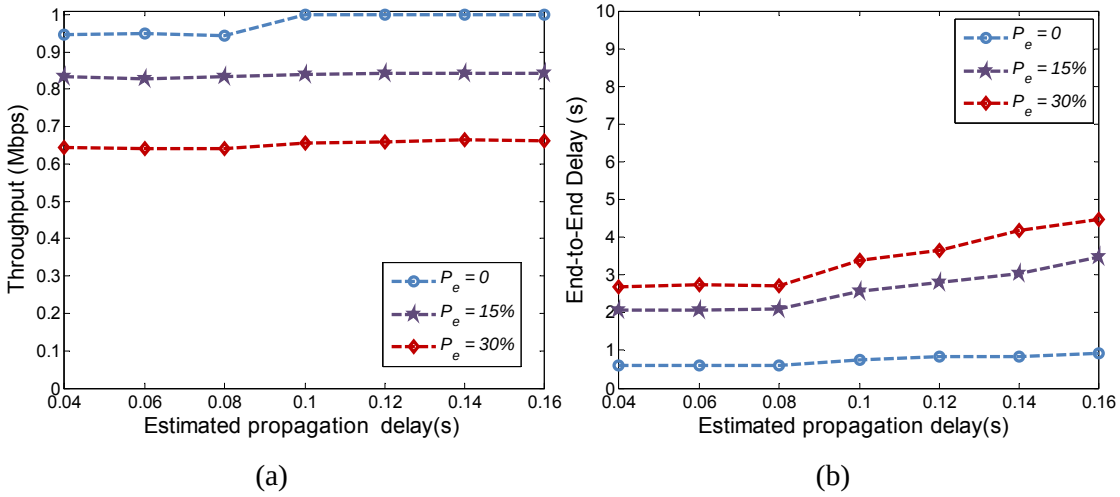

Figure 7: The effect of estimated propagation delay on (a)throughput and (b) end-to-end delay.

TCP/$\text{NC}_{\text{hop-by-hop}}$ decreases as losses increase, and this is due to that, the send window is large when P_e is small and there are more packets that are involved in the same encoding block, comparing with that when P_e is high. As a result, the time that packets wait to be decoded at the destination is longer when P_e is small.

6.2.2 Fairness, load balancing and stableness

In this part, the performance of fairness, load balancing and stableness of TCP-HONC in the whole system is studied. There are 4 TCP flows (denoted as flow 1-4) on node 1 that send packets to node 10 along the two red paths from time 0s and run for 1000s. In addition, there are also 4 TCP flows (denoted as flow 5-8) on node 2 that send packets to node 9 along the two gray paths from time 200s and run for 600s. It can be seen that hop (3, 5) is a bottleneck link. For each TCP flow, the throughput is calculated at intervals of 20s. The queue length on node 3 is also printed out at intervals of 2s. It is obvious that, when the network resource is used fairly, the ideal throughput for each TCP flow is equal to 0.5Mbps when there are only flow 1-4 and 0.375Mbps when there are flow 1-8.

As shown in Fig. 8 (c), the throughput of TCP flows in the network is close to the ideal throughput which means that the network resource is fairly utilized by TCP-HONC. The throughput of flow 1-8 fluctuates around the ideal throughput from time 200s to 800s and this is the effect when TCP-HONC balances traffic from congested paths to non-congested ones. TCP-HONC effectively balances traffic load among multiple paths.

The stableness of TCP-HONC is also important in practical application, especially when there are some bottleneck links in the network. In this simulation, hop (3, 5) is a bottleneck link. Therefore, the real-time queue length on node 3 reflects the stableness of TCP-HONC.

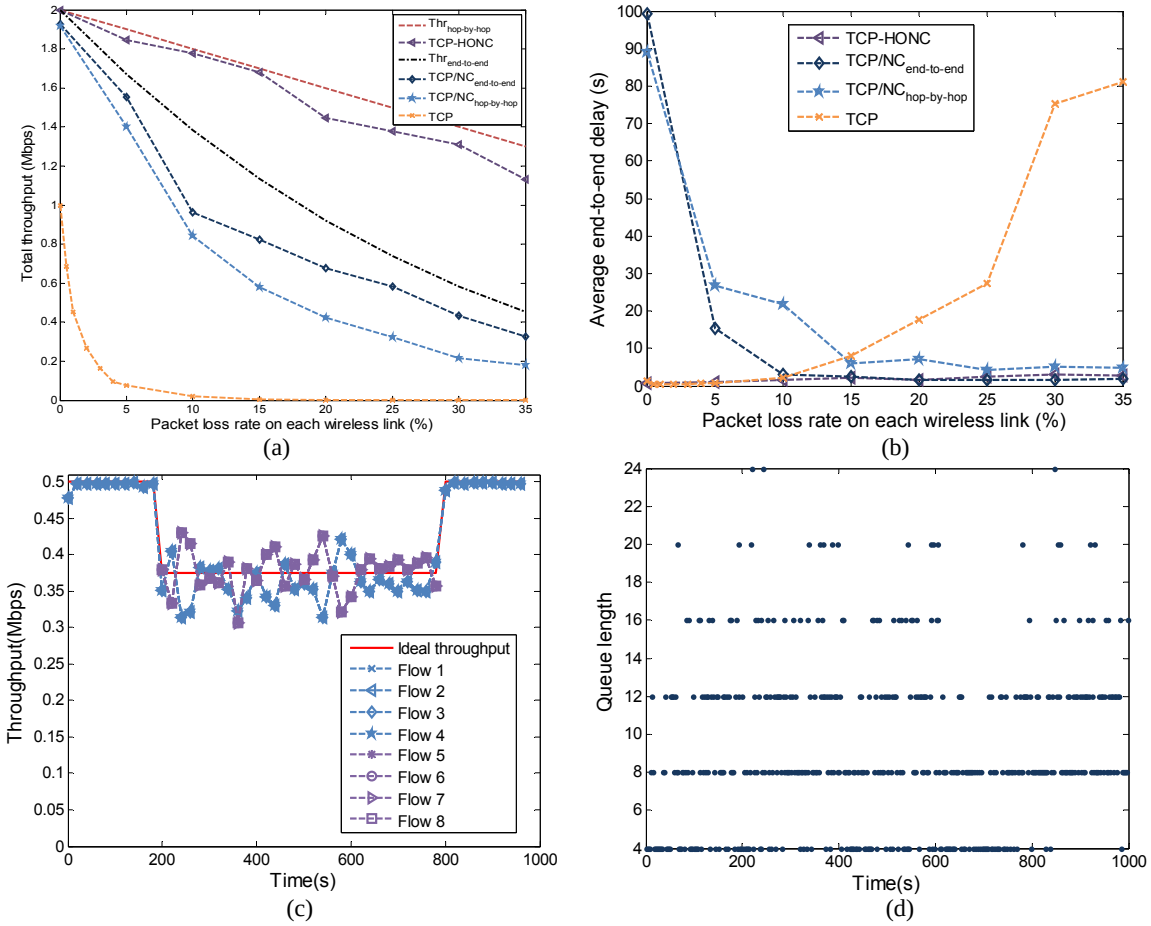

Figure 8: Results of multi-path scenario: (a) total throughput; (b) average end-to-end delay; (c) the effect of fairness and load balancing of TCP-HONC; (d) queue length on node 3.

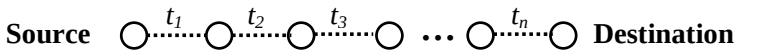

Figure 9: A single path unicast scenario with multiple lossy hops

As seen in Fig. 8 (d), the queue length on node 3 is very stable and is far smaller than the sum of all the TCP receive window.

7 Conclusion

In this paper, TCP-HONC is proposed to improve throughput and optimize end-to-end delay for TCP with NC in MRMC WMNs. It is based on the random linear network coding idea and performs HONC for TCP flows according to the local information of each mesh node. A delay optimization policy is introduced to reduce the end-to-end delay. With this policy, TCP-HONC adaptively adjusts the number of available encoding blocks and effectively schedules packets on each hop. Distributed rate control and path selection algorithm based on back-pressure method is also introduced to effectively utilize network resource and optimize end-to-end delay. Simulations show that TCP-HONC achieves a significant throughput gain. Meanwhile, the delay and delay jitter of TCP-HONC are relatively small even with a high loss rate. It outperforms TCP/$\text{NC}_{\text{end-to-end}}$ by 236% and TCP/$\text{NC}_{\text{hop-by-hop}}$ by 143% in throughput and reduces the average end-to-end delay to 27% and 90% in single path scenario, and it derives a total throughput which is 347% of the throughput of TCP/$\text{NC}_{\text{end-to-end}}$ in multi-path scenario, when the packet loss rate on each link is 35%. In addition, TCP-HONC derives a good fairness among all the TCP flows in simulation. Our approach is also capable to balance load from congested paths to non-congested ones and the queue length on the bottleneck link is also stable.

To be noticed, the throughput gain achieved by TCP-HONC is the result of re-encoding data at intermediate nodes and the achieved delay reduction benefits from one-hop ACK and retransmission scheme. TCP-HONC can be effectively used for applications in MRMC WMNs to get better performance.

Appendix

For a single path unicast scenario with multiple lossy hops, as illustrated in Fig. 9, let d be the propagation delay and p be the packet loss rate on each hop, then for the latency one the first hop, denoted as t_1, we have

$$t_1 = (1-p)d + p(2d + t_1). \tag{20}$$

Consequently, t_1 is given by

$$t_1 = \frac{d(1+p)}{1-p}. \tag{21}$$

For one-hop ACK and retransmission scheme, the transmission on each hop is similar, therefore, $Delay^n_{one-hop}$ is given by

$$Delay^n_{one-hop} = \sum_{i=1}^{n} t_i = nt_1 = \frac{nd(1+p)}{1-p}, \tag{22}$$

and $Delay^n_{one-hop} = nd$ when p = 0.

For end-to-end ACK and retransmission scheme, we have that

$$t_n = (1-p)d + p(d(n+1) + t_1 + t_2 + \cdots + t_{n-1}), \tag{23}$$

as a result, t_n is given by

$$t_n = \frac{d(1+np)}{1-p} + \frac{p(t_1 + t_2 + \cdots + t_{n-1})}{1-p}. \tag{24}$$

Next, we have

$$\begin{aligned} Delay^n_{end-to-end} &= \sum_{i=1}^{n} t_i \\ &= Delay^{n-1}_{end-to-end} + t_n \\ &= Delay^{n-1}_{end-to-end} + \frac{d(1+np)}{1-p} + \frac{p(t_1 + t_2 + \cdots + t_{n-1})}{1-p} \\ &= \frac{d(1+np)}{1-p} + \frac{Delay^{n-1}_{end-to-end}}{1-p}, \end{aligned} \tag{25}$$

$Delay^0_{end-to-end} = 0$, and $Delay^n_{end-to-end} = nd$ when $p = 0$.

Finally, by iteration, $Delay^n_{end-to-end}$ is given by

$$Delay^n_{end-to-end} = \frac{d(1+p^2)}{p(1-p)^n} - \frac{d}{p} + dp\sum_{i=0}^{n-2} \frac{n-i}{(1-p)^{i+1}}. \tag{26}$$

References

[1] Akyildiz, I.F.: 'A survey on wireless mesh networks', *IEEE Communications Magazine*, 2005, **43**, pp. S23-S30

[2] Jun, J.: 'The nominal capacity of wireless mesh networks', *Wireless Communications, IEEE*, 2003, **10**, (5), pp. 8-14

[3] 'Capacity of Wireless Mesh Networks', white paper, BelAir Networks, 2006.

[4] Lun, D. S., and Ho, T.: 'Network Coding: An Introduction'. Cambridge Univ. Press, 2008.

[5] Cheng, R.-S., Deng, D.-J., Huang, Y.-M., Huang,L., and Chao, H.-C.: 'Cross-Layer TCP with Bitmap Error Recovery Scheme in Wireless Ad Hoc Networks', *Telecommunication Systems*, 2010, **44**, (1), pp. 69-78.

[6] Aguayo, D., Bicket, J., Biswas, S.G., Judd, Morris, R.: 'Link-level measurements from an 802.11b mesh network', *ACM SIGCOMM Computer Communication Review*, 2004, **34**, (3), pp. 121-132

[7] Katti, S., Rahul, H., Hu, W., and Katabi, D.: 'XORs in the air: practical wireless network coding', *IEEE/ACM Transactions on Networking*, 2008, **16**, pp. 497-510

[8] Seferoglu, H., and Markopoulou, A.: 'I^{2}NC: Intra-and Inter-Session Network Coding for Unicast Flows in Wireless Networks'. INFOCOM, Shanghai, China, April 2011, pp. 1035-1043

[9] Hassayoun, S., and Maill, P.: 'On the impact of random losses on TCP performance in coded wireless mesh networks'. INFOCOM, San Diego, CA, USA March 2010, pp. 1-9

[10] Ahlswede, R., Cai, N., and Li, S.: 'Network information flow', *Information Theory, IEEE Transactions on*, 2000,**46**, (4), pp. 1204 - 1216

[11] Ho, T.: 'Networking from a network coding perspective'. PhD Thesis, Massachuaetts Institute of Technology, 2004

[12] Sundararajan, J.K., Shah, D., Medard, M., Mitzenmacher, M., and Barros, J.: 'Network Coding Meets TCP'. IEEE INFOCOM 2009, Rio de Janeiro, Brazil,Apr. 2009, pp. 280-288

[13] Sundararajan, J., Shah, D., and Mdard, M.: 'Network Coding Meets TCP: Theory and Implementation', *Proceedings of the IEEE*, 2011, **99**, (3), pp. 490-512

[14] Kim, M., Mdard, M., Barros, J.: 'Modeling Network Coded TCP Throughput: A Simple Model and its Validation'. Valuetools, Ecole Normale Suprieure de Cachan, France, May 2011, pp. 1-10

[15] Senger, C., Schober, S., Mao, T., and Zeh, A.: 'End - to - End Algebraic Network Coding for Wireless TCP / IP Networks'. ICT, Doha, Qatar, April 2010 pp. 607-612

[16] Gheorghiu S., and Toledo, A.: 'Multipath TCP with Network Coding for Wireless Mesh Networks'. ICC, Cape Town, South Africa, May 2010, pp. 3946-3950

[17] Sharma, V., Kalyanaraman, S., and Kar, K.: 'MPLOT: A transport protocol exploiting multipath diversity using erasure codes'. INFOCOM, Phoenix, AZ, USA, April 2008, pp. 121-125

[18] Sharma, V., Kalyanaraman, S., and Kar, K.: 'A multi-path transport protocol to exploit network diversity in airborne networks'. MILCOM, San Diego, CA, USA, November, 2008, pp. 1-7

[19] Sharma, V., Kalyanaraman, S., and Kar, K.: 'Exploiting multiple paths and diversity in wireless networks for high goodput and low latency'. COMSNETS, Bangalore, India, Jan. 2009, pp. 1-10

[20] Li, Y., Zhou, L., Yang, Y., and Chao, H.-C.: 'Optimization Architecture for Joint Multi-path Routing and Scheduling in Wireless Mesh Networks', *Mathematical and Computer Modelling*, 2011, **53**, (3-4), pp. 458-470

[21] Tang, Y.-L., Wu, T.-Y., Ding, J.-W., Chen, J.-J.: 'Resource Sharing and Bandwidth Allocation for WiMAX Mesh Networks Using Centralized Scheduling', *Journal of Internet Technology*, 2010, **11**, (2), pp.251-259

[22] Lun, D.S., Medard, M., and Effros, M.: 'On coding for reliable communi-cation over packet networks', *Physical Communication*, 2008, **1**, (1), pp. 3-20

[23] Lun, D.S., Muriel, M., and Koetter, R.: 'Further results on coding for reliable communication over packet networks'. Information Theory, Adelaide, SA ,Australia, September 2005, pp. 1848-1852

[24] Le, A., Member, S., Kum, D., and Cho, Y.: 'Routing with Load-Balancing in Multi-Radio Wireless Mesh Networks', *IEICE TRANS. COMMUN.*, 2009, **3**, pp. 700-708

[25] Sheikhan, M., Hemmati, E.: 'High reliable disjoint path set selection in mobile ad-hoc network using hopfield neural network', *IET Commun.*, 2011, **5**, (11), pp. 1566-1576

[26] Draves, R., Padhye, J., and Zill, B.: 'Routing in multiradio, multi-hop wireless mesh networks'. MobiCom '04, Philadelphia, PA, USA, September 2004, pp. 114-128

[27] Subramanian, A.P., Buddhikot, M.M., Miller, S.: 'Interference aware routing in multi-radio wireless mesh networks', WiMesh, Reston, Virginia, USA, Sept. 2006, pp. 55-63

[28] Tecnolgic, C. and Catalunya, D.T.D.: 'A Survey on Routing Protocols that really Exploit Wireless Mesh Network Features', *JOURNAL OF COMMUNICATIONS*, 2010, **5**, (3), pp. 211-231

[29] Tassiulas, L.: 'Adaptive back-pressure congestion control based on local information', *Automatic Control, IEEE Transactions on*, 1995, **40**, (2), pp. 236-250

[30] McCanne S., and Floyd, S.: 'ns Network Simulator'. [Online]. Available: http://www.isi.edu/nsnam/ns/, accessed Jan. 2012