CEDAR: A Low-latency and Distributed Strategy for Packet Recovery in Wireless Network

Chenxi Qiu*, Haiying Shen*, Sohraab Soltani†, Karan Sapra*, Hao Jiang⋆ and Jason Hallstrom⋆
*Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634, USA
⋆School of Computing, Clemson University, Clemson, SC 29634, USA
†Intelligent Automation, Inc., Rockville, MD 20855, USA

***Abstract*—Underlying link-layer protocols of wireless networks use the conventional "store and forward" design paradigm cannot provide highly sustainable reliability and stability in wireless communication, which introduce significant barriers and setbacks in scalability and deployments of wireless networks. In this paper, we propose a Code Embedded Distributed Adaptive and Reliable (CEDAR) link-layer framework that targets low latency and high throughput. CEDAR is the first comprehensive theoretical framework for analyzing and designing distributed and adaptive error recovery for wireless networks. It employs a theoretically-sound framework for embedding channel codes in each packet and performs the error correcting process in selected intermediate nodes in packet's route. To identify the intermediate nodes for the en/decoding for minimizing average packet latency, we mathematically analyze the average packet delay, using Finite State Markovian Channel model and priority queuing model, and then formalize the problem as a non-linear integer programming problem. Also, we propose a scalable and distributed scheme to solve this problem. The results from real-world testbed "NESTbed" and simulation with Matlab prove that CEDAR is superior to the schemes using hop-by-hop decoding and destination-decoding not only in packet delay but also in throughput. In addition, the simulation results show that CEDAR can achieve the optimal performance in most cases.**

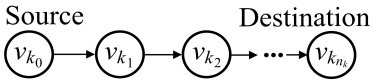
I. Introduction

Despite the unprecedented success and proliferation of wireless communication, there are major shortcomings in the underlying link-layer protocols in providing sustainable reliability and stability among wireless users. Popular wireless link-layer protocols, such as the retransmission ARQ or Forward Error Correction (FEC) based ARQ (HARQ) approaches (employed by the IEEE 802.xx and LTE standard suite) are designed to achieve some level of reliability by discarding a corrupted packet at the receiver and performing one or more retransmissions until the packet is decoded/received error-free or a maximum number of retransmission attempts is reached. This methodology suffers from degradation of throughput and overall system instability since decoding failures at the receiver due to a small number of bit errors lead to packet drops and discarding a large number of correctly delivered data bits.

Many leading research efforts [1]–[11] have highlighted the inefficiencies of these link-layer protocols and proposed a variety of remedy solutions. The majority of these efforts either consider variations of the ARQ, HARQ or a hybrid approach of both schemes [1], [5], [11], [12]. They largely follow the traditional "store-and-forward" link-layer design paradigm: each data packet must be fully received and corrected by every relay node before it is forwarded. This design paradigm increases stability but still cannot provide high stability due to its hop-by-hop operation.

It is our belief that achieving the ultimate objective of the development of ubiquitous and heterogeneous wireless networks demands fundamental and radical changes to the conventional link-layer protocol design. Thus, we study and develop alternative optimal and low-complexity error recovery strategies in link-layer design to achieve high reliability and stability by partially and optimally selecting relay nodes. The objectives of the strategies are to ensure, (1) Low end-to-end latency and rapid delivery of packets; (2) High throughput with minimum data loss. To meet these objectives, we develop solutions that address the following key issues: (1) *Minimizing propagation and transmission (prop&tran) delay*: at which intermediate nodes (if any) a link-layer packet should be detected to minimize packet delay? (2) *Minimizing queuing delay*: as multiple relay nodes in a route perform error recovery on the same packet stream and one node may perform error recovery for multiple packet streams, how to select relay nodes that provides global reliability and stability in a wireless network with many source-destination packet streams?

As a solution, we develop mathematical models for the prop&tran delay and queuing delay for a packet based on the path length between two consecutive decoding nodes in a route (route segment length). Through rigorous mathematical analysis on the models, we derive two propositions that (1) can identify the intermediate nodes for decoding which minimize prop&tran delay of a packet, and (2) prove that balanced en/decoding load distribution among decoding nodes in the network minimizes the queuing delay. Based on the propositions, we formulate the problem of minimizing delay as a non-linear integer programming problem. However, due to the NP-hard nature of the problem and impracticability of collecting all required information to find the global optimal solution, we propose a sub-optimal Code Embedded Distributed Adaptive and Reliable (CEDAR) link-layer framework for wireless networks. CEDAR is a distributed and cooperative error recovery design, which provides an adaptive environment for various error recovery strategies with respect to reliability and stability. We summarize our contributions as follows: (1) We build a model for the probability of decoding failure of a packet traveling through a given number of hops based on the Finite State Markovian Channel (FSMC) model; (2) We build rigid mathematical models for the prop&tran delay, and queuing delay for a packet; (3) We formalize the problem of choosing the intermediate en/decoing nodes for minimum delay as a non-



ARQ, HARQ: retransmission at each hop
ACE: packet recovery at each hop
CEDAR: packet recovery at selected hops

Fig. 1. Protocols for packet recovery.

linear integer programming problem which is an NP-hard problem; (4) We propose a distributed sub-optimal strategy for CEDAR that achieves high reliability and stability.

The reminder of the paper is organized as follows. Section II states the problem that needs to be solved to minimize the packet delay. Section III introduces mathematical model that formalizes the problem as a non-linear integer programming problem and derives two propositions to minimize the delay. Guided by the propositions, Section IV details the design of CEDAR for solving the problem in Section II, and Section V presents performance evaluation of CEDAR in comparison with previous schemes. Section VI presents a review of the related works. The final section concludes with a summary of contributions and a discussion on future research work.

II. Problem Statement

Consider a wireless network comprised of N *nodes* represented as $\mathcal{V} = \{v_1, v_2, ..., v_N\}$. Each traffic flow from a *source node* to a *destination node* transverses over a predetermined set of *links* (a route specified by the network layer). Let $\mathcal{R} = \{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_B\}$ denote the transmission routes. Each route $\mathbf{r}_k$ $(1 \leq k \leq B)$ carries a data stream following Poisson distribution with arrival rate λ_k. A vector of nodes $\mathbf{r}_k = \{v_{k_0}, v_{k_1}, ..., v_{k_{n_k}}\}$ $(v_{k_0}, v_{k_1}, ..., v_{k_{n_k}} \in \mathcal{V})$ represents the nodes in $\mathbf{r}_k$, where $n_k = |\mathbf{r}_k|$.

As shown in Fig. 1, to reach the destination, each packet flow needs to travel through all nodes in the predetermined route, and some of these nodes are responsible for en/decoding the packets. In the ARQ and HARQ protocols [1], [5], [11], each hop drops distorted packets and requests for complete or partial retransmission of the original packets. Though these methods guarantee the reliability between any pair of nodes, they cause high delays and low throughput due to numerous retransmissions at every hop. CEDAR introduces a new flexible environment for link-layer error recovery: the error correction process is performed in a distributed manner where selected (and not all) intermediate nodes participate in performing error recovery. The key problem in CEDAR is how to identify candidates among the intermediate nodes for the en/decoding process to optimally decrease latency and increase throughput over the entire network.

First, we build a model to calculate the prop&tran delay $(D^i_{\mathrm{p\&t}}(\mathbf{n}'_i))$, and a model to calculate the queuing delay $(D_{\mathrm{q}}(\mathbf{n}'_i))$ based on the lengths of the routing paths $(\mathbf{n}'_i)$ of the packets crossing an intermediate node v_i. We use these models to minimize the expected delays and ultimately identify the positions of intermediate nodes for en/decoding in each route in CEDAR. Throughout the paper, we use the key terms provided in the following definitions:

Definition 1 (*Key node*): A key node of route $\mathbf{r}_k$ is a node responsible for en/decoding the packets traveling along $\mathbf{r}_k$. Matrix $\mathcal{X} = (x_{i,k})_{N\times B}$ denotes whether v_i is a key node in $\mathbf{r}_k$:

$$x_{i,k} = \begin{cases} 1, & v_i \text{ is the key node in } \mathbf{r}_k; \\ 0, & v_i \text{ is not the key node in } \mathbf{r}_k. \end{cases} \quad (1)$$

Definition 2 (*Route segment*): A route segment of $\mathbf{r}_k$ is a section of the end-to-end path between one key node to either the endpoints or another key node.

In each route segment, the packet sender (the first key node) encodes the packets and the packet receiver (the second key node) decodes the packets. In other words, the second key node is responsible for decoding for its route segment. Use $n'_{i,k}$ to denote the number of hops in a route segment with decoding node v_i in $\mathbf{r}_k$. Let $\mathbf{n}'_i$ denote the group of the lengths of route segments responsible by v_i, i.e., $\mathbf{n}'_i = \{n'_{i,1}, n'_{i,2}, ..., n'_{i,B}\}$. $n'_{i,k} = 0$ if v_i has no responsibility of decoding the packet in $\mathbf{r}_k$. Let $\lambda'_{i,k}$ denote the arrival rate of the data stream that v_i is responsible for en/decoding in $\mathbf{r}_k$. Then $\lambda'_{i,k} = \lambda_k \times x_{i,k}$. We use $\overline{D(\mathbf{n}'_i)}$ to denote the total average delay when one packet crosses v_i and use 0-1 variable $f_{i,k}$ to denote whether v_i is in $\mathbf{r}_k$. If yes, $f_{i,k} = 1$; otherwise $f_{i,k} = 0$.

Objective. The objective of CEDAR is to minimize the total delay of the packets in the entire system, which can be represented as:

$$\min \sum_{i=1}^{N} \left(\overline{D(\mathbf{n}'_i)} \sum_{k=1}^{B} \lambda'_{i,k} \right)$$
$$\text{s.t. } x_{i,k} \leq f_{i,k} \quad (2)$$

The packet delay in v_i (which is composed of prop&tran and queueing delays) is a function of $\mathbf{n}'_i$. This will be deduced in the mathematical analysis in Section III. We use $\overline{D_{\mathrm{p\&t}}(\mathbf{n}'_i)}$ to denote the average prop&tran delay of all the packet streams being decoded in v_i, and use $\overline{D_{\mathrm{q}}(\mathbf{n}'_i)}$ to denote the average queuing delay of the packet stream in v_i. Then, the total average delay when one packet crosses v_i is:

$$\overline{D(\mathbf{n}'_i)} = \overline{D_{\mathrm{q}}(\mathbf{n}'_i)} + \overline{D_{\mathrm{p\&t}}(\mathbf{n}'_i)} \quad (3)$$

Thus, we need to minimize $\overline{D_{\mathrm{q}}(\mathbf{n}'_i)}$ and $\overline{D_{\mathrm{p\&t}}(\mathbf{n}'_i)}$ in order to achieve the objective of CEDAR in Formula (2). To this end, in Sections III-A and III-B, we model the Bit Error Rate (BER) fluctuations of wireless channels and probability of successful decoding. Sections III-C and III-D use this model to formulate the prop&tran delay $\overline{D_{\mathrm{p\&t}}(\mathbf{n}'_i)}$ and the queuing delay $\overline{D_{\mathrm{q}}(\mathbf{n}'_i)}$. Finally, Section III-E derives two propositions to minimize $\overline{D_{\mathrm{q}}(\mathbf{n}'_i)}$ and $\overline{D_{\mathrm{p\&t}}(\mathbf{n}'_i)}$, respectively. Guided by the propositions, we design CEDAR in Section IV.

III. Mathematical Modeling

In this section, we first present a Markovian wireless channel model to capture the variations in wireless error conditions due to non-stationary wireless noise and calculate BER of a packet when it goes through several channels. Using this model, we analyze the relationship between the number of hops a packet crosses and the probability of its successful decoding. This relationship leads us to calculate the prop&tran delay and queuing delay, respectively. By minimizing the two delays, we can find the locations of intermediate nodes in a route for decoding. Finally, we formulate the problem of minimizing the sum of the delays as a non-linear integer programming problem. The analytical results and the formed problem lay the foundation for the

design of an optimized strategy for choosing intermediate nodes for the CEDAR packet recovery.

A. Markovian Channel Model

Finite State Markovian Channel model (FSMC) [13] is a channel model that uses finite state Markov chain to describe the process, under which errors are introduced into a transmitted packet over a wireless route. The model has a finite set of error states $\mathcal{S}$= (s_1, s_2, ..., s_K) ($|\mathcal{S}| = K$), each corresponding to a Binary Symmetric Channel (BSC). The channel model can be considered as a combination of K number of various BSCs with unique BERs (ϵ) (i.e., $\epsilon_l \neq \epsilon_j$ for $l \neq j$, $l, j = 1, 2, ..., K$). Assuming packets are transmitted during discrete time slots τ_i ($i = 1, 2, 3...$) which can be referred as *transmission intervals*. During the i^{th} transmission interval, a packet is transmitted from a BSC to another BSC with cross-over BER ϵ_i. Each ϵ_i of a particular τ_i is valued from $\mathcal{S}$. The Markovian model assumes a homogenous and stationary Markov chain with transition probability matrix $\mathcal{T} = (t_{ij})_{K\times K}$ and initial probability $\pi = (\pi_{\mathbf{1}}, ..., \pi_{\mathbf{2}})$. $\mathcal{T} = (t_{ij})_{K\times K}$ can be trained on real channel traces by using the statistics of previous transmission intervals. This captures the effects of multi-path fading and interferences on the channel BER in every transmission interval using a single aggregated model [13]. The system average BER can be calculated as:

$$\bar{\epsilon} = \sum_{k=0}^{K-1} \pi_k \epsilon_k \tag{4}$$

Based on this prior work, we calculate the average BER for consecutive wireless links within a route segment in a cascaded system, and derive Lemma 3.1.

Lemma 3.1: The BER in a cascade system where a node travels along links with states $s_{a_1}s_{a_2}...s_{a_n}$ ($1 \leq a_1, a_2, ..., a_n \leq K$) can be given by:

$$\overline{\epsilon^n} \approx \sum_{\{s_{a_1}...s_{a_n}\}\in\mathcal{S}^n} \left(\pi_{a_1} \prod_{j=1}^{n-1} t_{a_j a_{j+1}} \sum_{i=1}^{n} \epsilon_i \right) \tag{5}$$

where $\mathcal{S}^n$ represents all the possible set of series which is composed of n elements and each element is contained in S (notice each series can have duplicated elements).

Proof: Let

$$\mathcal{E}_i = \begin{bmatrix} 1-\epsilon_i & \epsilon_i \\ \epsilon_i & 1-\epsilon_i \end{bmatrix} \tag{6}$$

be the transition probability matrix when the packet's channel is in s_i. We then derive

$$\mathcal{E}_i = \mathcal{B}^{-1} \begin{bmatrix} 1 & 0 \\ 0 & 1-2\epsilon_i \end{bmatrix} \mathcal{B} \tag{7}$$

where

$$\mathcal{B} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{8}$$

Then, we consider the situation that one bit goes through the cascade of n nodes and the bit's channel state is changed in the sequence of $s_{a_1}, s_{a_2}, ...s_{a_i}, ..., s_{a_n}$ ($1 \leq a_i \leq K, 1 \leq i \leq n$). In this case, the transition probability matrix through n nodes, denoted as $\mathcal{E}_{a_1a_2...a_n}$, is given by

$$\begin{aligned} \mathcal{E}_{a_1a_2...a_n} &= \mathcal{E}_{a_1}\mathcal{E}_{a_2}...\mathcal{E}_{a_n} = \mathcal{B}^{-1} \begin{bmatrix} 1 & 0 \\ 0 & \prod_{i=1}^{n}(1-2\epsilon_{a_i}) \end{bmatrix} \mathcal{B} \\ &= \begin{bmatrix} \frac{1+\prod_{i=1}^{n}(1-2\epsilon_{a_i})}{2} & \frac{1-\prod_{i=1}^{n}(1-2\epsilon_{a_i})}{2} \\ \frac{1-\prod_{i=1}^{n}(1-2\epsilon_{a_i})}{2} & \frac{1+\prod_{i=1}^{n}(1-2\epsilon_{a_i})}{2} \end{bmatrix} \end{aligned}$$

Thus, the BER of the cascade of n nodes (a_1, a_2, a_3, ..., a_n) equals:

$$\epsilon_{a_1a_2...a_n} = \frac{1-\prod_{i=1}^{n}(1-2\epsilon_{a_i})}{2} \tag{9}$$

The probability that such a aforementioned situation occurs equals:

$$\Pr[X = \{s_{a_1}s_{a_2}...s_{a_n}\}] = \pi_{a_1} \prod_{j=1}^{n-1} t_{a_j a_{j+1}} \tag{10}$$

where X is a random variable represents the series. Then, the expectation of error bit through the cascade of n hops is given by:

$$\overline{\epsilon^n} = \sum_{\{s_{a_1}...s_{a_n}\}\in\mathcal{S}^n} \Pr[X = \{s_{a_1}s_{a_2}...s_{a_n}\}] \times \epsilon_{a_1a_2...a_n} \tag{11}$$

$$= \sum_{\{s_{a_1}...s_{a_n}\}\in\mathcal{S}^n} \pi_{a_1} \prod_{j=1}^{n-1} t_{a_j a_{j+1}} \frac{1-\prod_{i=1}^{n}(1-2\epsilon_{a_i})}{2} \tag{12}$$

When ϵ_1, ϵ_2, ..., $\epsilon_n \ll 1$

$$\overline{\epsilon^n} \approx \sum_{\{s_{a_1}...s_{a_n}\}\in\mathcal{S}^n} \left(\pi_{a_1} \prod_{j=1}^{n-1} t_{a_j a_{j+1}} \sum_{i=1}^{n} \epsilon_{a_i} \right) \tag{13}$$

■

B. Probability of Successful Decoding

CEDAR is developed based on the error recovery mechanism in the ACE Communication Model [14]. Thus, we first introduce ACE before we present the mathematical models. Specifically, during τ_i, a transmitter encodes data symbols z_i with parity codes x_i (referred as type-I parity code) to create a codeword $C_i(z_i, x_i)$. It transmits a packet $M_i = (C_i(z_i, x_i), y_i)$, where y_i denotes the additional parity (hereafter type-II parity) symbols for recovering previously received corrupted packets at the receiver. We also use x_i, y_i and z_i to denote the number of their symbols. The receiver utilizes x_i to decode C_i. If the decoding operation fails, the receiver stores C_i in its buffer and issues a request along with ACK_i for more parity symbols. The transmitter then sends additional parity $y_j (j > i)$ along with M_j. We use $m_i = x_i + y_i$ to denote the total number of parity symbols of M_i.

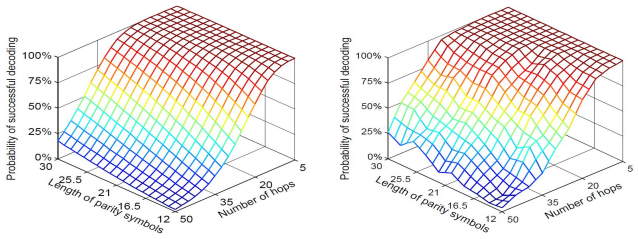
First, consider a simple cascade model ($v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_n$) in which a packet stream goes through a series of nodes $v_0, v_1, v_2, ..., v_n$ and is encoded and decoded at v_0 and v_n, respectively. We can approximate Equ. (5) by Equ. (14) to calculate the BER for a routing through n nodes under the Markovian channel model:

$$\overline{\epsilon^n} \approx n\bar{\epsilon} \tag{14}$$

As ACE, we take Reed-Solomon codes [15] as an example, which is a kind of non-binary cyclic error-correcting codes, for channel coding. In the Reed-Solomon codes, each symbol is composed of b bits, indicating that the probability of error for each symbol equals:

$$\overline{\epsilon^{n,b}} = 1 - (1-\overline{\epsilon^n})^b \tag{15}$$

The number of error symbols introduced in one packet M_i with a length of $z_i + m_i$ symbols through n hops can be represented by a random variable E_i following a binomial distribution $E_i \approx Bi\left(z_i + m_i, \epsilon^{n,b}\right)$. If the error estimate is $\hat{\epsilon}^{n,b}$ for one symbol of b bits, the receiver is capable of correcting up to αm_i errors out of $|C_i|$ symbols in packet M_i, where α is a function measuring the expected error-correcting capability of a particular decoder based on $\hat{\epsilon}^{n,b}$. For instance, the error-correcting capability of the Reed-Solomon codes is half as many as redundant symbols (i.e.,



(a) The surface of $F_\alpha(n, m_i)$ (b) Simulation results

Fig. 2. Comparison of results.

$\alpha = 0.5$) [15]. The probability of successfully recovering data bits by a parity code with m_i length symbols equals:

$$P^i_{\text{succ}} = \Pr[E_i \le \alpha m_i] \tag{16}$$

$$= \sum_{k=0}^{\lfloor \alpha m_i \rfloor} \binom{z_i + m_i}{k} \left(1 - \hat{\epsilon}^{n,b}\right)^{z_i + m_i - k} \left(\hat{\epsilon}^{n,b}\right)^k \tag{17}$$

From Equ. (16), we observe that P^i_{succ} is a discrete function of two variables:

$$P^i_{\text{succ}} = F_\alpha(n, m_i). \tag{18}$$

$F_\alpha(n, m_i)$ is monotonically decreasing function of n (number of hops in a route), and is monotonically increasing function of m_i (number of symbols in parity code). This is observed in Fig. 2. Fig. 2 (a) shows the surface of $F_\alpha(n, m_i)$ when $\alpha = 0.5$, $z_i = 20$ and $\hat{\epsilon} = 1.5 \times 10^{-3}$, when n is varied from 5 to 50, and m_i is varied from 12 to 30. Fig. 2 (b) shows the simulation results of the successful decoding rate under the FSMC model between a source and a destination node with n hops between them (n is ranged from 5 to 50). The consistency between the analysis results and simulation results verifies Equ. (16) and Equ. (18).

Based on Equ. (18), the number of times (i.e., trials) a packet is required to be decoded until it is recovered has a nonhomogeneous geometric distribution (denoted by G) [16] given that the length (i.e., number of symbols) of predetermined parity code equals m_t at the t^{th} trial.

Lemma 3.2: We use $f^t_G(n)$ to denote the probability of successful decoding on the t^{th} decoding trial for one packet going through n hops. Then,

$$f^t_G(n) = F_\alpha(n, m_t) \times \prod_{i=1}^{t-1} (1 - F_\alpha(n, m_i)) \tag{19}$$

C. Propagation and Transmission Delay

In this section, we consider the prop&tran delay of a packet M_i. We use $D^{i,t}_{\text{p\&t}}(n)$ to denote the prop&tran delay of M_i when the parity code of M_i has been transmitted for t times through n nodes. Let $D_{\text{p}}(n)$ represent the propagation delay for one packet going through n nodes and $D_{\text{ACK}}(n)$ denote the transmission delay of the ACK packet. Further, let $D^{i_k}_{\text{t}}(n)$ denote the transmission delay of the packet M_{i_k}. The length of this packet is $L^{i_k}_{\text{pac}} = z_{i_k} + m_{i_k}$, where M_{i_k} is the k^{th} packet that carries M_i's parity symbols for the k^{th} time after k-1 times of recovery failures (i.e., type-II parities). Then, as Fig. 3 shows, $D^{i,t}_{\text{p\&t}}(n)$ can be calculated as

$$D^{i,t}_{\text{p\&t}}(n) = \sum_{k=0}^{t-1} \left(2D_{\text{p}}(n) + D_{\text{ACK}}(n) + D^{i_k}_{\text{t}}(n)\right) + D_{\text{prop}}(n) + D^{i_t}_{\text{t}}(n) \tag{20}$$

We use R to denote the bandwidth provided to the route M_i travels. Assume electric signal travels at velocity c in the media and the distance of each hop (d) is an invariable. Then, $D_{\text{ACK}}(n)$, $D^{i_k}_{\text{t}}(n)$ and $D_{\text{p}}(n)$ can be calculated as:

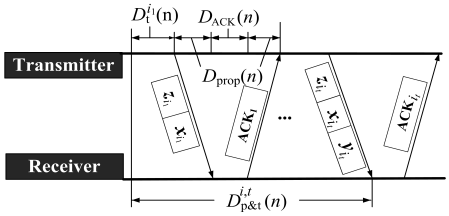


Fig. 3. Transmission and propagation delay.

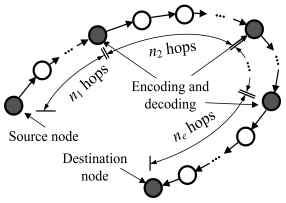


Fig. 4. Route segment.

$$D_{\text{ACK}}(n) = \frac{L_{\text{ACK}}}{R} \times n, \; D^{i_k}_{\text{t}}(n) = \frac{L^{i_k}_{\text{pac}}}{R} \times n \tag{21}$$

$$D_{\text{prop}}(n) = \frac{d}{c} \times n \tag{22}$$

Based on Equ. (20), (21), and (22), we can derive that:

$$D^{i,t}_{\text{p\&t}}(n) = n \left[\sum_{k=0}^{t-1} \left(\frac{2d}{c} + \frac{L_{\text{ACK}} + L^{i_k}_{\text{pac}}}{R}\right) + \frac{d}{c} + \frac{L^t_{\text{pac}}}{R}\right] \tag{23}$$

Based on Equ. (19) in Lemma 3.2 and Equ. (23), we retrieve Lemma 3.3 for the expectation of $D^i_{\text{p\&t}}(n)$.

Lemma 3.3: The expected propagation and transmission delay of a packet M_i $D^i_{\text{p\&t}}(n)$ can be calculated by:

$$\overline{D^i_{\text{p\&t}}(n)} = \sum_{t=1}^{\infty} f^t_G(n) D^{i,t}_{\text{p\&t}}(n) \tag{24}$$

As shown in Fig. 4, given a route from a source node to a destination node, we can divide the route into e segments, each segment having length of $n_1, n_2, ..., n_e$. In each route segment, a packet is encoded at the first node and decoded at the last node. The goal of our scheme is to determine the $n_1, n_2, ..., n_e$ in order to minimize the prop&tran delay of a packet from the source to the destination, i.e., to achieve

$$\begin{aligned} \min \quad & \textstyle\sum_{j=1}^{e} \overline{D^i_{\text{p\&t}}(n_j)} \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{e} n_j = n \end{aligned}$$

D. Queuing Delay

In a priority queuing model, packets entering a buffer are classified into several different priority categories and added into different queues accordingly. The packets with lower priority can enter the server only when all queues for higher priority queues are empty. In the wireless network, for any single node v_i that is responsible for decoding U routes, there will be U poisson streams $(\lambda_1, \lambda_2, ..., \lambda_U)$ arriving at this node. v_i needs to decide the order of arriving packets to decode. Thus, by regarding v_i as the server in the model, we can use the priority queuing model (M/M/1/∞/∞/PR) [17] for analyzing the queuing delay. Note when a packet fails to decode, it will be decoded (i.e., join in a queue) again when it received another type-II parity code along with another packet. In order to balance the queuing delay of each node, we propose a strategy for determining the priority of decoding packets. That is, the more times a packet has failed

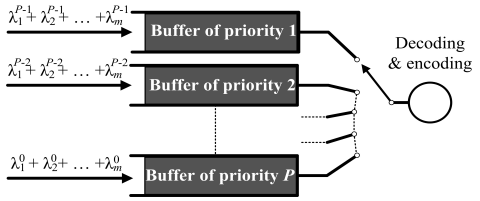


Fig. 5. Priority queue.

to be corrected, the higher priority it will be given when it is re-decoded. When a packet suffers P number of failures, it is dropped. We do not consider the stream of retransmission for packets after P failures because the probability of failing more than P times is extremely small.

Fig. 5 gives a sketch of the priority queuing model in our scheme. In the figure, $\lambda_1^p, \lambda_2^p, ..., \lambda_U^p$ denote the arriving rate of the streams whose packets are re-decoded at the $(p-1)^{\text{th}}$ time. Recall that if a packet fails to decode, it is stored in the buffer waiting for the next parity symbol for recovery. The re-decoded streams, which are "generated" by failed decoded packets, follow Poisson distribution [18] and their arrival rates satisfy the following condition:

$$\lambda_k^p\left(n_k'\right) = \lambda_k^1 \prod_{t=1}^{p-1}\left(1 - F_\alpha\left(n_k', m_t\right)\right) \tag{25}$$

where n_k' $(1 \le k \le U)$ denotes the number of hops in the route segment where k^{th} traffic stream has traveled through since its last en/decoding in the route. Assuming that λ_i^1 and m_t have been pre-determined, the value of $\lambda_k^p\left(n_k'\right)$ is determined by n_k'. We assign priority p to the packet stream of λ^p $(p = 1, 2, ..., P)$. The packets in a queue with the highest priority P enter the server (decoding and encoding part) first. If the queue of priority P is empty, then the packets of priority $P-1$ enter the server; and so on.

Assume there are U data streams in the p^{th} $(1 \le p \le P)$ priority queue, because each packet stream follows Poisson distribution, all of these streams can be combined into one stream $\lambda^p = \sum_{j=1}^{U} \lambda_j^p$. We use ρ_l to represent the *utilization* of a server when the first packet in the buffer with priority l enters the server and use Y_l to represent service time for a packet in a queue with priority l [17]. Recall $\mathbf{n}'$ is the set of all n_k' $(1 \le k \le U)$. Then, $\rho_l\left(\mathbf{n}'\right)$ can be calculated as

$$\rho_l\left(\mathbf{n}'\right) = \overline{Y_l} \times \sum_{j=1}^{U} \lambda_j^l\left(n_j'\right) \tag{26}$$

W_l represents the average delay of packets with priority l packets and W_0 represents the average delay for one tagged waiting packet due to a packet already in service. W_0 can be calculated as:

$$W_0\left(\mathbf{n}'\right) = \sum_{l=1}^{P} \frac{\overline{Y_l^2}}{2\overline{Y_l}} \times \rho_l\left(\mathbf{n}'\right) \tag{27}$$

As a result, the waiting time for each of packets is:

$$W_p\left(\mathbf{n}'\right) = \frac{W_0\left(\mathbf{n}'\right) + \sum_{l=p+1}^{P} \rho_l\left(\mathbf{n}'\right) W_l\left(\mathbf{n}'\right)}{1 - \sum_{l=p}^{P} \rho_l\left(\mathbf{n}'\right)} \tag{28}$$

From Kleinrock's conservation theorem in priority queuing model [19], the expected queuing delay for one packet in any node can be calculated as:

$$\overline{W_{\text{que}}\left(\mathbf{n}'\right)} = \sum_{p=1}^{P} \rho_p\left(\mathbf{n}'\right) W_p\left(\mathbf{n}'\right) = \frac{\rho\left(\mathbf{n}'\right) W_0\left(\mathbf{n}'\right)}{1 - \rho\left(\mathbf{n}'\right)} \tag{29}$$

where

$$\rho\left(\mathbf{n}'\right) = \sum_{p=1}^{P} \rho_p\left(\mathbf{n}'\right) \tag{30}$$

Now, we consider the queueing delay for one packet, which might enter the queueing system several times due to re-en/decoding. During time interval T (T is large enough), the total number of packets N_{total} that enter the queueing system equals:

$$N_{\text{total}}\left(\mathbf{n}'\right) = \sum_{l=1}^{P} \sum_{j=1}^{U} \lambda_j^l\left(n_j'\right) \times T \tag{31}$$

The total waiting time can be given by:

$$W_{\text{total}}\left(\mathbf{n}'\right) = N_{\text{total}}\left(\mathbf{n}'\right) \times \overline{W_{\text{que}}\left(\mathbf{n}'\right)} \tag{32}$$

Lemma 3.4: The expectation of the total queuing time for one packet when it goes through a node with $\mathbf{n}_i' = \{n_{i,1}', n_{i,2}', n_{i,3}', ..., n_{i,U}'\}$ can be calculated as

$$\begin{aligned} \overline{D_{\text{q}}\left(\mathbf{n}'\right)} &= \frac{W_{total}\left(\mathbf{n}'\right)}{\sum_{k=1}^{U} \lambda_k^1\left(n_k'\right)} \\ &= \frac{\sum_{p=1}^{P} \sum_{k=1}^{U} \lambda_k^p\left(n_k'\right) \times \overline{W_{\text{que}}\left(\mathbf{n}'\right)}}{\sum_{k=1}^{U} \lambda_k^1} \end{aligned} \tag{33}$$

E. Minimizing the Delays

As shown in Section II, we need to minimize $\overline{D_{\text{q}}\left(\mathbf{n}_i'\right)}$ and $\overline{D_{\text{p\&t}}\left(\mathbf{n}_i'\right)}$ in order to achieve the objective of CEDAR in Formula (2). According to Equ. (24), the prop&tran delay of the packet stream for $\mathbf{r}_k$ in v_i is calculated as:

$$\overline{D_{\text{p\&t}}\left(n_{i,k}'\right)} = \sum_{t=1}^{\infty} f_G^t\left(n_{i,k}'\right) D_{\text{p\&t}}^t\left(n_{i,k}'\right) \tag{34}$$

Consequently, the average prop&tran delay of all the packet stream decoded in v_i is calculated as:

$$\overline{D_{\text{p\&t}}\left(\mathbf{n}_i'\right)} = \frac{\sum_{k=1}^{U}\left[\lambda_{i,k}' \sum_{t=1}^{\infty} f_G^t D_{\text{p\&t}}^t\left(n_{i,k}'\right)\right]}{\sum_{k=1}^{U} \lambda_{i,k}'} \tag{35}$$

where $\mathbf{n}_i' = \{n_{i,1}', n_{i,2}', ..., n_{i,U}'\}$. The average queuing delay of the packet stream in v_i can be derived from Equ. (33):

$$\overline{D_{\text{q}}\left(\mathbf{n}_i'\right)} = \frac{\sum_{i=1}^{P} \sum_{j=1}^{U} \lambda_j^l\left(n_j'\right) \times \overline{W_{\text{que}}\left(\mathbf{n}_i'\right)}}{\sum_{j=1}^{U} \lambda_j^1\left(n_j'\right)} \tag{36}$$

By minimizing the above $\overline{D_{\text{q}}\left(\mathbf{n}_i'\right)}$ and $\overline{D_{\text{p\&t}}\left(\mathbf{n}_i'\right)}$, we retrieve two propositions presented below.

1) Minimizing Queuing Delay

Definition 3 (*En/Decoding load*): The en/decoding load of v_i, denoted by q_i, is defined as the sum of the arrival rates for all the packet streams that v_i is responsible for en/decoding. That is,

$$q_i = \sum_{k=1}^{U} \lambda_{i,k}' \tag{37}$$

The en/decoding load is used as a metric to determine the *key nodes* for each route. When determining the key nodes for a new route, the system should avoid choosing nodes with relatively high en/decoding load.

Proposition 3.1: Suppose there are N nodes $\{v_1, v_2, ..., v_N\}$ and B routes $\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_B$ with total arrival rate $\sum_k^B \lambda_k' = \lambda_{\text{total}}'$, each packet is required to be decoded once in its route, and each node can decode the packet in any route, and also $\mathbf{n}_1' = \mathbf{n}_2' = ... = \mathbf{n}_N'$. To minimize the total queueing delay for all the packets, the en/decoding load for each node should be the same. That is:

$$\lambda_i' = \frac{\lambda_{\text{total}}'}{N} \quad (i = 1, 2, 3, ..., N) \tag{38}$$

Proof: According to Equ. (26), Equ. (27), Equ. (29) and (30), the queuing delay of packets decoded at v_{k_i} can be derived as:

$$\overline{D_{\mathrm{q}}^{i}(\mathbf{n}')} = \frac{\rho(\mathbf{n}')}{1-\rho(\mathbf{n}')} \frac{\sum_{l=1}^{P}\sum_{k=1}^{B}\lambda_k^{'l}(n'_{i,k})}{\sum_{k=l}^{B}\lambda_k^{'1}} W_0 = \frac{AJ\lambda_i'^2}{1-J\lambda_i'} \tag{39}$$

where A and J are invariants figured by Equ. (30), Equ. (26) and Equ. (25). Then, the following equation can be derived:

$$\sum_{i=1}^{N}\frac{AJ\lambda_i'^2}{1-J\lambda_i'} = -AC\lambda'_{\mathrm{total}} + \frac{AN}{J} + \frac{A}{J}\sum_{i=1}^{N}\frac{1}{(1-J\lambda_i')} \tag{40}$$

Let $H_i = 1 - J\lambda_i'$ and then $\sum_{i=1}^{N} H_i = N - J\lambda'_{\mathrm{total}}$. According to Cauchy−Schwarz inequality [18], we find that:

$$\sum_{i=1}^{N}\frac{N-J\lambda'_{\mathrm{total}}}{H_i} = \sum_{i=1}^{N}\frac{1}{H_i}\sum_{i=1}^{N}H_i \tag{41}$$

$$= \sum_{i=1}^{N}\frac{1}{\left(\sqrt{H_i}\right)^2}\sum_{i=1}^{N}\left(\sqrt{H_i}\right)^2 \geq \left(\sum_{i=1}^{N}\sqrt{\frac{1}{H_i}}\sqrt{H_i}\right)^2 = N^2 \tag{42}$$

from which, we can derive:

$$\sum_{i=1}^{N}\frac{1}{(1-J\lambda_i')} = \sum_{i=1}^{N}\frac{1}{H_i} \geq \frac{N^2}{N-J\lambda'_{\mathrm{total}}} \tag{43}$$

and then derive:

$$\sum_{i=1}^{N}\frac{AJ\lambda_i'^2}{1-J\lambda_i'} \geq -A\lambda'_{\mathrm{total}} + \frac{AN}{J} + \frac{AN^2}{J(N-\lambda'_{\mathrm{total}})} \tag{44}$$

which reaches its minimum value when $1-J\lambda_1' = 1-J\lambda_2' = ... = 1-J\lambda_N'$, or the values of λ_i' $(i = 1, 2, .., N)$ are equal to each other. ■

According to Proposition 3.1, given several packet streams and number of nodes required to decode these packets, we need to balance the en/decoding load for all of these nodes.

2) Minimizing Prop&Tran Delay

Consider a route with n hops that is not affected by the interference from any other routes. Our objective is to divide it into several route segments with the size n_1, n_2, ..., n_e respectively in order to minimize the total delay of packet stream transmission. We consider one of these route segment that has n_k hops, and use $\overline{D_{\mathrm{p\&t}}^{\mathrm{ave}}}$ to denote the average prop&tran delay for each hop in this route segment. That is:

$$\overline{D_{\mathrm{p\&t}}^{\mathrm{ave}}(n_i)} = \frac{\overline{D_{\mathrm{p\&t}}(n_i)}}{n_i} \ (i = 1, 2, 3, ..., e) \tag{45}$$

where $\overline{D_{\mathrm{p\&t}}^{\mathrm{ave}}(n_i)}$ is a function of n_i. We use $\overline{D_{\mathrm{p\&t,min}}^{\mathrm{ave}}}$ to represent the minimized value of $\overline{D_{\mathrm{p\&t}}^{\mathrm{ave}}(n_i)}$, and we need to search n_{opt} that satisfies $\overline{D_{\mathrm{p\&t}}^{\mathrm{ave}}(n_{\mathrm{opt}})} = \overline{D_{\mathrm{p\&t,min}}^{\mathrm{ave}}}$,

Proposition 3.2: To divide one route into several route segments, the optimal length for each route segment should be n_{opt} in order to minimize the prop&tran delay for the packet delivery.

Proof: The sum of prop&tran delay for the e route segments equals:

$$\sum_{i=1}^{e}\overline{D_{\mathrm{p\&t}}(n_i)} = \sum_{i=1}^{e}\frac{\overline{D_{\mathrm{p\&t}}(n_i)}}{n_i} \times n_i$$

$$\geq \overline{D_{\mathrm{p\&t,min}}^{\mathrm{ave}}} \times \sum_{i=1}^{e} n_i = \overline{D_{\mathrm{p\&t,min}}^{\mathrm{ave}}} \times n \tag{46}$$

When $n_1 = n_2 = ... = n_e = n_{\mathrm{opt}}$, $\sum_{i=1}^{e}\overline{D_{\mathrm{p\&t}}(n_i)}$ reaches its minimum value. ■

IV. Scalable and Distributed Scheme

The objective function of CEDAR in Formula (2) is a nonlinear integer programming problem. Solving this problem leads to minimizing the total delay for all the packets in the network. This, however, requires each node collect a global knowledge of the network including the routes and the arrival rate for each traffic stream, which is nearly impractical in wireless applications such as wireless ad hoc networks. Even though the global knowledge is available, the problem is NP-hard as it is a nonlinear integer programming problem [20]. Thus, we need to design a scalable and distributed scheme for identifying the key nodes for each route. Fortunately, the two propositions in Section III-E provide foundation for the design of a distributed scheme to select key nodes.

Simply put, Proposition 3.1 indicates that the scheme should try to balance the en/decoding load of each node to minimize the queuing delay; and Proposition 3.2 indicates the optimal route segment length (i.e., the positions of key nodes) to minimize the prop&tran delay. If both requirements can be satisfied simultaneously, the scheme will satisfy the objective function. However, these two requirements may conflict with each other. We identify different network traffic load situations that each proposition should be primarily considered, and also propose a method to coordinately consider these two propositions when choosing key nodes.

Algorithm 1: Identify key nodes in route **r** executed by each node in **r** in a light-traffic network.

```
begin
    Set SEN_FIN, REC_FIN and DEC to 0 ;
    while SEN_FIN = 0 or REC_FIN = 0 do
        Listen to other nodes;
        if it has received ACK_REC from the next node in r then
            SEN_FIN ← 1;
        if it receives (OPT_HOP, FLAG) from the previous node in
        r then
            REC_FIN ← 1;
            Send ACK_REC to the previous node;
            if FLAG = 0 then
                DEC ← 1 // It is a key node;
                FLAG ← OPT_HOP;
            else
                FLAG ← FLAG - 1;
```

Case I (light traffic): When a wireless network has light traffic, because the influence from queuing delay is much less significant, we mainly consider the prop&tran delay. As Proposition 3.2 indicates, we first search the value of n_{opt} and then set OPT_HOP $= n_{\mathrm{opt}}$, and set FLAG $=$ OPT_HOP. In a routing algorithm [21], every node keeps a routing table, and a source node sends out a message to find the route to a destination for transmitting a packet stream. After a source-destination route has been discovered, each node in the route determines whether it is a key node in a distributed manner by executing the key node identification algorithm. Algorithm 1 presents the pseudo code of this algorithm executed by every node (except source node and destination node), say v, in a route **r** in the case of light network traffic. Here, SEN_FIN presents whether v has received ACK from the next node in **r**; REC_FIN presents whether v has received (OPT_HOP, FLAG) from the

previous node in **r**; DEC presents whether v is responsible for en/decoding;

Case II (heavy traffic): When a wireless network has heavy traffic, we aim to balance the en/decoding load for each node through the route to reduce queuing delay while reducing the prop&tran delay. When a new route is built, the CEDAR scheme first executes Algorithm 1. Simultaneously each node along the route piggybacks its en/decoding load to the packet, and the last node sends the information back. Then the source node knows the series of nodes identified as "potential key nodes" and their en/decoding loads, and calculates the average en/decoding load through the route, denoted as AVE_LOAD. It then checks whether the en/decoding load of each identified key node is larger than a pre-defined threshold (AVE_LOAD + BOUND), where BOUND is a predetermined value. An overloaded potential key node probes its nearby nodes sequentially until finding a node with load within the threshold or meeting an identified potential key node. The pseudocode of this algorithm is presented in Algorithm 2.

Algorithm 2: Select key nodes with consideration of load balance in a heavy-traffic or normal-traffic network.

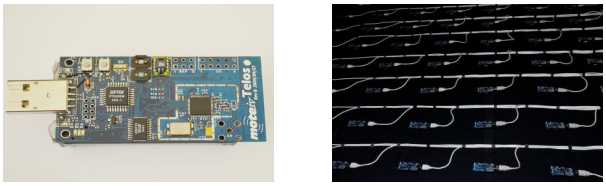
```
begin
    Use Algorithm 1 to get the "potential key nodes" in route r;
    Let node i be one node selected as the "potential key node";
    j = 0;
    while j ≤ ⌊OPT_HOP/2⌋ do
        if LOAD_{i+j} ≤ LOAD_{i−j} then
            if LOAD_{i+j} ≤ (BOUND + AVE_LOAD) then
                return i + j;
        else
            if LOAD_{i−j} ≤ (BOUND + AVE_LOAD) then
                return i − j;
        j = j + 1;
    return 0;
```

V. Performance Evaluation

This section presents the results of experiments on NESTbed [22] and simulation with MATLAB. We compared CEDAR with the global optimal solution (OPTIMAL), the traditional link-layer protocols, where packet is decoded hop by hop [6], [14] (HBH), and with another solution where packet is only decoded at the destination (DEST). In order to evaluate the effect of the load balancing algorithm (Algorithm 2) in CEDAR, we also test the performance of CEDAR without this algorithm denoted by CEDAR*.

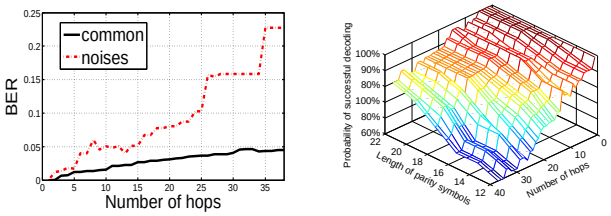
A. Experiments on Real-World NESTbed

NESTbed is an open testbed for developing wireless sensor systems [22]. It is a collection of 80 TELOSB sensors (Fig. 8 (a)) that are arranged in a grid (Fig. 8 (b)). The sensors have CC2420 Chip and communicate using the IEEE 802.15.4 standard. We verify our mathematical models and evaluate the performance of CEDAR on NESTbed. We created a multi-hop network of TELOSB sensors running Tiny-OS 2.1.0 written in NESC. We use Reed-Solomon codes to detect and fix errors in a packet, if the number of error bits exceeds the capability of Reed-Solomon codes for correction, the receiver ask for retransmission.



(a) TELOSB sensors (b) NESTbed

Fig. 6. Experiment.



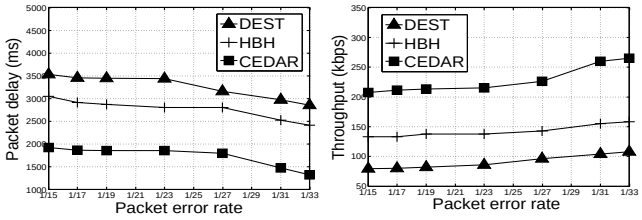
(a) BER in multiple hops (b) Successful decoding rate

Fig. 7. Comparison of results.

1) Mathematical model verification: We measured BER after the packet traveled for different numbers of hops in two scenarios, *common* and *noises*. Fig. 7 (a) shows BER versus the number of hops. We see that BER increases as the number of hops increases in both scenarios. This is because as the number of hops increases, more flipped bits are generated and errors tend to be cumulative and propagated along a routing path in the multi-hop network. Also, *common* produces much lower BER than *noises* as more noises increase BER.

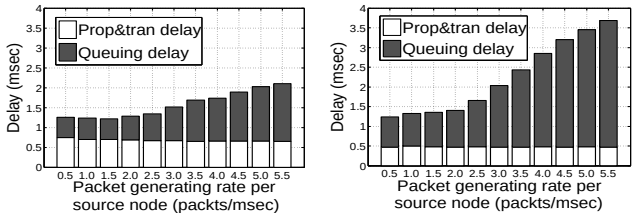
We then measured the probability of successful decoding when the error correction node is away from the source node by different number of hops and when the parity symbols have different lengths. We used the Reed-Solomon codes as the error correcting codes that have error-correcting capability of $\alpha = 0.5$ [10] and prime polynomial of x^3+x+1 [15]. Fig. 7 (b) shows the probability of successful decoding in a multi-hop network with varying parity symbol lengths and the number of hops between the error correction node and the source node. The figure illustrates that if the error correction node is further away from the source, the probability of a packet drop increases. Also, when the length of parity symbols increases, the probability of successful decoding increases, which is consistent with Formula (16) (Fig. (a)).

2) Scheme performance evaluation: In the experiments on NESTbed, we chose 8 sources and 8 destinations and the source-destination path length was 75 hops. Each source node generated 500 packets at a time interval varying from 80ms to 160ms. We measured the delay of transmissions and the throughput, which is defined as the total size of all the packets divided by the total time used for transmitting all the packets. The packet error rate is defined as the average percent of unsuccessfully transmitted packets in each hop. To test the performance of the three schemes in different environments, we manually changed the packet error rate from 1/15 to 1/33. Fig. 8 (a) shows the packet delay of CEDAR, DEST and HBH. We find that the average packet delay of CEDAR is much lower than that of DEST and HBH. DEST has the highest delay because it assigns the decoding work to each packet's destination rather than the intermediate nodes, which generates much higher probability

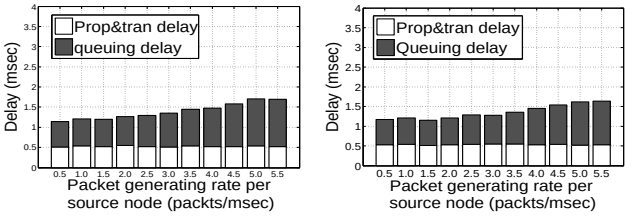


(a) Packet delay (b) Throughput

Fig. 8. Experimental results on real-world NESTbed.



(a) DEST (b) HBH



(c) CEDAR* (d) CEDAR

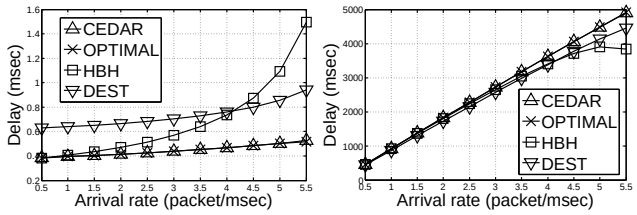
Fig. 9. Prop&tran delay and queuing delay.

of packet re-decoding due to higher probability of packet errors, thus increasing the delay. The delay of HBH is higher than that of CEDAR because HBH requires packets to be en/decoded in each hop, which generates high en/decoding load on intermediate nodes, leading to high queuing delay. Fig. 8 (b) shows the throughput of three schemes. From the figure, we can find that the throughput follows CEDAR $>$ HBH $>$ DEST. This is because lower packet transmission delay usually leads to higher throughput in the network.

B. Simulation on Matlab

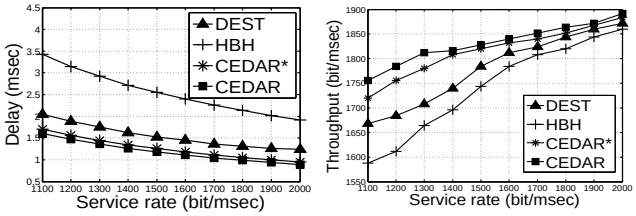
We conducted simulation on Matlab to evaluate the performance of CEDAR. We built a 9×9 grid network with each node located in one grid and randomly selected 16 pairs of source node and destination node. Each packet contains 20 data symbols, 5 type-I parity symbols and 5 type-II parity symbols. Each symbol has 5 bits. Also, we randomly chose nodes connecting each pair of source node and destination node as the route.

Fig. 9 (a), (b), (c) and (d) compare prop&tran delay and queuing delay computed by HBH, DEST, CEDAR* and CEDAR respectively. From the figure we can find that: (1) the queuing delay increases as the generating rate of each data stream increases but the prop&tran delay remains nearly constant; (2) the queuing delay increases more significantly in HBH than in DEST and CEDAR (i.e., it follows CEDAR $<$ DEST $<$ HBH); (3) for prop&tran delay it follows HBH $<$ CEDAR $<$ DEST, (4) CEDAR generates the same prop&tran delay but lower queuing delay than CEDAR*, and (5) the total packet delay follows CEDAR $<$ CEDAR* $<$ DEST $<$ HBH. For (1), this is because queuing delay is determined by the generating rate of the source node



(a) Average packet delay (b) Throughput

Fig. 10. Packet delay and throughput.



(a) Average packet delay (b) Throughput

Fig. 11. Packet delay and throughput (with OPTIMAL).

but the prop&tran delay is independent of it. For (2), (4) and (5), HBH has higher queuing delay since it generates more en/decoding load on intermediate nodes. In contrast to HBH, DEST only assigns the decoding work to each packet's destination, which increases both prop&tran delay and queuing delay due to higher probability of packet re-decoding (as Equ. (18) shows). Instead of accumulating decoding work on the destinations, CEDAR* and CEDAR chooses a number of intermediate nodes to be responsible for the en/decoding work to reduce the probability of re-decoding. CEDAR performs better than CEDAR* because CEDAR distributes the en/decoding load of the intermediate nodes more evenly, which reduces the queuing delay as indicated in Proposition 3.1

Fig. 11 (a) compares the average packet delay of HBH, DEST, CEDAR* and CEDAR with various service rates. The results is consistent with the results in Fig. 9. Also, we find that CEDAR produces higher throughput than CEDAR*, DEST and HBH in Fig. 11 (b). This is because lower packet transmission delay usually leads to higher throughput in the network. Fig. 10 (a) and (b) compares OPTIMAL with CEDAR, CEDAR*, DEST and HBH in terms of *packet delay* and *throughput*. Considering NP-hard feature of the problem, we only set a small scale network (6 source nodes and 6 destination nodes). The results demonstrate that CEDAR can achieve almost the "best" performance in terms of packet delay even in the distributed manner.

VI. Related Work

The link-layer protocol of the current TCP/IP stack has adopted variations of error recovery mechanisms to provide reliability for point-to-point communication especially for wireless systems. Different wireless communication standards currently utilize variations of error control protocols that generally can be categorized into ARQ [12] and HARQ-based [5], [11] protocols. For instance IEEE802.11 WiFi uses ARQ where a receiving node discards corrupted packets (even when there is only a single bit error) and requests for a retransmission. The 4G/LTE deploys HARQ with Turbo Codes where the sender node encodes the packet payload

using Turbo channel codes [23] prior to the transmission. Accordingly, the receiver node requests for a retransmission when the decoding of the received packet fails. In conjunction with the current wireless link-layer standards, there is significant work and research conducted to imporve the performance of either ARQ- or HARQ-based protocols. Several kinds of HARQ protocols (see [5], [11] and the reference therein) improve the throughput of the ARQ schemes by packet combining, e.g. by keeping the erroneous received packets and utilizing them for detection and packet recovery. Examples of recent efforts for combating the inefficiency of ARQ-based wireless protocols include Partial Packet Recovery (PPR) [9], Cross-Layer Design with Side-information (CLDS), and Automatic Code Embedding (ACE) framework [14]. Some of these approaches, such as PPR and SOFT, exploit physical layer information regarding the quality of individual bits to increase the probability of recovering corrupted packets. Other schemes, such as CLDS and ACE, utilize information available in the current 802.11 link-layer protocols in conjunction with error correcting codes to recover corrupted packets. Ilyas *et al.* [24] proposed the "Poor Man's SIMO System" (PMSS) to reduce packet losses in networks of commodity IEEE 802.15.4 sensor motes using cooperative communication and diversity combination. Based on mathematical analysis, Jelenkovi *et al.* [4] proposed a new dynamic packet fragmentation algorithm that can adaptively match channel failure characteristics. These aforementioned works have significantly improved the ARQ- and HARQ-based link-layer performance and provide a comprehensive error control approach for wireless communication. However virtually all of these efforts follow the conventional TCP/IP link-layer "store-and-forward" design paradigm where each relay node verifies the correctness of each packet before forwarding it to the next node. This inherently introduces substantial overhead on bandwidth utilization and throughput and the overall end-to-end delay.

VII. Conclusion

In this paper, our objective is to find an optimal solution to choose immediate nodes in transmission routes for en/decoding packets in wireless networks in order to minimize the packet delay and increase the throughput. We mathematically analyze the packet delay and model the problem as an integer programming problem, which helps to discover a globally optimal solution. Taking into account the scalability of the network and limitation of the information that each node can collect, we propose a distributed scheme that can achieve performance comparable to the globally optimal solution. The simulation results in MATLAB demonstrates that our scheme performs better than previous packet recovery schemes. In our future work, we aim to use random graph to analyze wireless networks (e.g., mobile ad-hoc network) where the network topology varies frequently for accurate formulation of the stochastic of the network.

Acknowledgements

This research was supported in part by U.S. NSF grants OCI-1064230, CNS-1249603, CNS-1049947, CNS-1156875, CNS-0917056 and CNS-1057530, CNS-1025652, CNS-0938189, CSR-2008826, CSR-2008827, Microsoft Research Faculty Fellowship 8300751, and U.S. Department of Energy's Oak Ridge National Laboratory including the Extreme Scale Systems Center located at ORNL and DoD 4000111689.

References

[1] R. Cohen and L. Grebla, G.and Katzir, "Cross-layer hybrid fec/arq reliable multicast with adaptive modulation and coding in broadband wireless networks," in *Proc. of INFOCOM*, 2009.

[2] Z. Guo, J. Huang, and et al., "A practical joint network- channel coding scheme for reliable communication in wireless networks," in *Proc. of MobiHoc*, 2009.

[3] G. Woo, P. Kheradpour, D. Shen, and D. Katabi, "Beyond the bits: Cooperative packet recovery using physical layer information," in *Proc. of MOBICOM*, 2007.

[4] P. R. Jelenkovi and J. Tan, "Dynamic packet fragmentation for wireless channels with failures," in *Proc. of MobiHoc*, 2008.

[5] E. C. Strinati, S. Simoens, and J. Boutros, "Performance evaluation of some hybrid arq schemes in ieee 802.11a networks," in *Proc. of VTS*, 2003.

[6] K. C. Lin, N. Kushman, and D. Katabi., "Harnessing partial packets in 802.11 networks," in *Proc. of MOBICOM*, 2008.

[7] S. S. Karande and H. Radha, "Non-linear integer programming by darwin and boltzmann mixed strategy," *IEEE Transactions On Multimedia*, 2008.

[8] M. Ghaderi, D. Towsley, J. Kurose, U. of Calgary, and Calgary, "Reliability gain of network coding in lossy wireless networks," in *Proc. of INFOCOM*, 2008.

[9] K. Jamieson and H. Balakrishnan, "PPR: Partial packet recovery for wireless networks," in *Proc. of SIGCOMM*, 2007.

[10] S. Soltani, K. Misra, and H. Radha, "Delay constraint error control protocol for real-time video communication," *IEEE Transaction on Multimedia*, 2009.

[11] H. Yomo, S. S. Chakraborty, and R. Prasad, "PHY and MAC performance evaluation of IEEE 802.11a WLAN over fading channels," *IEEE Transaction on Multimedia*, 2009.

[12] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications.* NJ: Prentice-Hall: Englewood Cliffs, 2004.

[13] H. S. Wang and N. Moayeri, "Finite-state markov channel - a useful model for radio communication channels," *IEEE Transactions on vehicular technolgoy*, 1995.

[14] S. Soltani, K. Misra, and H. Radha, "On link-layer reliability and stability for wireless communication," in *Proc. of MOBICOM*, 2008.

[15] S. Wicker and V. Bhargava, "Institute of electrical and electronics engineering, inc," *IEEE Press*, 1994.

[16] M. Mandelbaum, M. Hlynka, and P. H. Brill, "Nonhomogeneous geometric distributions with relations to birth and death processes," *Springer J. TOP Business Econ.*, 2007.

[17] J. L. Hammond and P. J. O'Reilly, *Performance Analysis of Local Computer Networks*. Reading, Massachusetts: Addison-Wesley Publishing Company, 1988.

[18] S. Chahramant, *Fundamentals of Probability with Stochastic Process*. Reading, Massachusetts: Western New England College, 1986.

[19] V. B. Iversen, *Teletraffic Engineering Handbook*. Technical University of Denmark, 2001.

[20] P. Tian, J. Ma, and D.-M. Zhang, "Non-linear integer programming by darwin and boltzmann mixed strategy," *European Journal of Operational Research*, 1996.

[21] L. L. Peterson and B. S. Davie, *Computer network: a system approach*. Morgan Kaufmann, 2007.

[22] A. R. Dalton and J. O. Hallstrom, "An interactive, source-centric, open testbed for developing and profiling wireless sensor systems," *International Journal of Distributed Sensor Networks*, March 2009.

[23] D.N.Rowitch and L.B.Milstein, "On the performance of hybrid fec/arq systems using rate-compatible low-density parity-check codes." *ITC*, 2000.

[24] M. Ilyas, M. Kim, and H. Radha, "Reducing packet losses in networks of commodity IEEE 802.15.4 sensor motes using cooperative communication and diversity combination," in *Proc. of INFOCOM*, 2009.