Delay-Optimal Broadcast for Multihop Wireless Networks Using Self-Interference Cancellation

Xinyu Zhang and Kang G. Shin, *Fellow*, *IEEE*

Abstract—Conventional wireless broadcast protocols rely heavily on the 802.11-based CSMA/CA model, which avoids interference and collision by conservative scheduling of transmissions. While CSMA/CA is amenable to multiple concurrent unicasts, it tends to degrade broadcast performance significantly, especially in lossy and large-scale networks. In this paper, we propose a new protocol called *Chorus* that improves the efficiency and scalability of broadcast service with a MAC/PHY layer that *allows* packet collisions. Chorus is built upon the observation that packets carrying the same data can be effectively detected and decoded, even when they overlap with each other and have comparable signal strengths. It resolves collision using symbol-level interference cancellation, and then combines the resolved symbols to restore the packet. Such a collision-tolerant mechanism significantly improves the transmission diversity and spatial reuse in wireless broadcast. Chorus' MAC-layer cognitive sensing and scheduling scheme further facilitates the realization of such an advantage, resulting in an asymptotic broadcast delay that is proportional to the network radius. We evaluate Chorus' PHY-layer collision resolution mechanism with symbol-level simulation, and validate its network-level performance via ns-2, in comparison with a typical CSMA/CA-based broadcast protocol. Our evaluation validates Chorus's superior performance with respect to scalability, reliability, delay, etc., under a broad range of network scenarios (e.g., single/multiple broadcast sessions, static/mobile topologies).

Index Terms—Optimal broadcast, wireless ad hoc and mesh networks, collision resolution, multipacket reception, self-interference cancellation, analog network coding

1 Introduction

Network-wide broadcast is a fundamental primitive for many communication protocols in multihop wireless networks, such as route discovery and information dissemination. An efficient broadcast protocol needs to deliver a packet (or a continuous stream of packets) from the source node to all other nodes in the network, with high packet-delivery ratio (PDR) and low latency. To improve PDR in a lossy network, multiple relay nodes can forward and retransmit each packet, thereby creating retransmission diversity. To reduce latency and resource usage, however, the number of transmissions must be kept to minimum, since redundant retransmissions waste channel time, slowing down the packet's delivery to the edge of the network. Therefore, a delicate balance needs to be maintained between PDR and delay.

To date, efficient broadcast support, in the form of theoretical analysis [1], [2], [3] or practical protocol design [4], has mostly focused on the CSMA/CA MAC-layer scheduling model. CSMA/CA has proven to be an effective distributed scheduling scheme, especially via the 802.11 family of MAC standards. The limitation of CSMA/CA, however, has not been examined carefully in case of network-wide broadcast. While its fine-tuned sensing and scheduling reduces collision, CSMA/CA inevitably misses transmission opportunities, lowering channel usage and spatial reuse. This problem becomes acute, especially for network-wide broadcast with latency constraints.

Fig. 1a illustrates a typical scenario where CSMA/CA restricts the broadcast efficiency. With CSMA/CA, the delivery of one packet from source S to all other nodes requires at least three time slots. A and B cannot transmit concurrently, even if they have to forward the same packet. Suppose node D in a lossy network had already received the packet, while C and E await its retransmission from A and B, respectively. An optimal scheduling protocol would be oblivious of the collision at D, and allow A and B to transmit the packet concurrently. However, this is not possible in CSMA/CA, as one of them will defer its transmission immediately upon sensing the other's activity.

In this paper, we introduce a novel broadcast protocol, called *Chorus*, based on a MAC layer that adopts *CSMA with collision resolution* (CSMA/CR). Chorus is built upon the key insight that *packets carrying the same data can be detected and decoded, even when they overlap at the receiver with comparable strength*. With Chorus, collision of the same packets from different relays can be effectively resolved. The advantage of such a collision-tolerant protocol is obvious, as shown in Fig. 1b. With collision resolution, A and B can now transmit packets immediately and independently after receiving them from the source. Node D exploits Chorus' collision resolution to decode the two collided packets from A and B. Therefore, only two time slots are required to deliver one packet over the entire network, due to the improved *spatial reuse*. Moreover, when links are unreliable, the two decoded packets from A and B create *transmit diversity* for the common receiver D, without consuming any additional channel time.

• *The authors are with the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, The University of Michigan, 2260 Hayward St., Ann Arbor, MI 48109-2121.*
E-mail: {xyzhang, kgshin}@eecs.umich.edu.

Manuscript received 30 Dec. 2010; revised 14 Sept. 2011; accepted 6 Oct. 2011; published online 20 Oct. 2011.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2010-12-0593.
Digital Object Identifier no. 10.1109/TMC.2011.233.

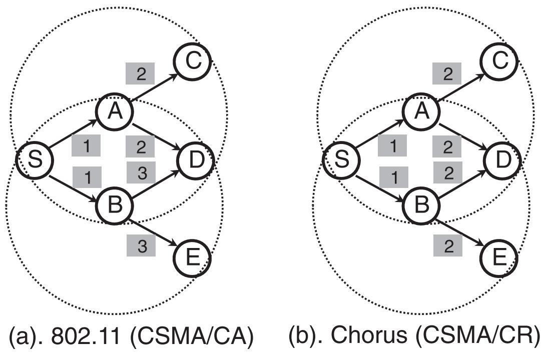


Fig. 1. Broadcast with traditional CSMA/CA in 802.11 in comparison with Chorus' CSMA/CR (CSMA with collision resolution). The shaded tags denote the order of transmissions.

Both the spatial reuse and the transmit diversity gain in Chorus are realized via its collision resolution scheme, which is based on *self-interference cancellation* [5]. Unlike traditional transmit diversity schemes such as beamforming [6], Chorus requires neither symbol time synchronization nor instantaneous channel state information. In reality, it is difficult to synchronize the independent transmitters A and B at the symbol level [6]. Chorus exploits the asynchrony between them to identify collision-free symbols in the overlapping packets. It then initiates an iterative cancellation process that subtracts clean and known symbols from the collided ones, and obtains estimates of unknown symbols. The decoding succeeds as long as one packet has sufficient SNR, hence realizing the diversity offered by multiple transmitters.

At the MAC layer, Chorus adds a *cognitive sensing and scheduling* module to the 802.11 CSMA mechanism. Specifically, senders back off only when they sense a packet on the air that has a different identity from what they intend to transmit. Such a cognitive MAC allows Chorus to fully exploit the advantage of collision resolution, while maintaining friendliness to background traffic. In addition, the collision-resolution capability enables anonymous broadcast at the network layer, without any topology or neighborhood information.

To quantify the effectiveness of Chorus, we establish an analytical framework for its achievable SNR and bit error rate (BER), which takes into account the error-propagation effects in iterative collision resolution. We further analyze its network-level performance in terms of latency and throughput. With a joint design of CSMA/CR and broadcast, Chorus achieves $\Theta(r)$ latency (r is the network radius), which is asymptotically optimal and unachievable in existing CSMA/CA-based broadcast protocols.

To verify the feasibility of Chorus' collision resolution, we implement the iterative decoding and packet combination in a symbol-level simulator. To evaluate Chorus's performance in large networks, we feed the above fine-grained analytical and simulation results into the PHY layer of ns-2, implement the CSMA/CR MAC, and broadcast protocol, and compare Chorus with a CSMA/CA-based protocol. In a large set of randomly chosen topologies, Chorus is shown to make several-fold performance improvement in latency and PDR. The performance gain is relatively insensitive to network size, source rate, and link quality, and is observed for both static and mobile topologies, and in both single- and multisource broadcast scenarios. These salient properties are important, especially for information dissemination in large-scale wireless networks, and represent the importance of exploiting PHY-layer signal processing to improve application performance.

The remainder of this paper is organized as follows: In Section 2, we review existing work in comparison with Chorus. We introduce the collision resolution mechanism in Section 3, and then the cognitive sensing, scheduling, and network-layer broadcast scheme in Section 4. In Section 5, we derive Chorus' achievable SNR and BER, and analyze its asymptotic broadcast performance. We evaluate Chorus' performance via simulation in Section 6, and conclude the paper in Section 7.

2 Related Work

Efficient broadcast in multihop wireless networks has been studied extensively, from both theoretical and practical perspectives. From the theoretical perspective, it is well known that scheduling a minimum-latency broadcast is NP-hard, either in a general undirected graph [3] or in a unit disk graph (UDG) [1]. Without the minimum latency constraint, analytical solutions demonstrated the feasibility of scheduling with time complexity $\Omega(r \log n)$ [7] in a distributed anonymous broadcast, and $r + O(\log r)$ [2] in centralized broadcast with a known topology, where r and n denote the network radius and number of nodes. More recent work has improved the efficiency, and adopted more realistic models, such as the interference graph [8].

The above algorithmic solutions generally assume perfect MAC-layer scheduling. In reality, scheduling in wireless networks is mostly based on distributed CSMA/CA. The widely used 802.11 standards [9] provide best-effort service broadcast, using CSMA/CA without any ACK or retransmissions. Practical broadcast protocols have mostly adopted the 802.11 CSMA/CA and extended it to multihop networks. A main mechanism is to prune the topology, leaving only a backbone that covers the entire topology. The double-coverage broadcast (DCB) [4], for example, reduces redundant transmissions by selecting nodes that cover more neighbors, while ensuring each node is covered at least twice, such that retransmission can be exploited to improve delivery ratio. The fundamental difference between Chorus and such existing protocols lies in its MAC layer scheduling protocol. With a joint design of CSMA/CR and network level broadcast, Chorus can achieve the $\Theta(r)$ latency bound, hence it has both theoretical and practical relevance.

Network coding is another approach that improves the broadcast reliability for wireless networks [10]. It simplifies the relay-selection problem by allowing random mixing of information at relays. However, network coding only applies to continuous broadcast, where a batch of source packets can be encoded. Network coding improves broadcast efficiency for lossy and mobile networks, but still cannot achieve the optimal latency in general [10].

The advent of high-performance software radios (SRs) has inspired wireless protocols beyond the CSMA/CA paradigm. For instance, the classical concept of interference cancellation [5] in information theory has been realized in SR-based wireless LANs [11]. Interference cancellation resolves overlapped packets by first decoding the one with stronger RSS, treating the weak packet as noise, and then subtracting the decoded strong packet, thus obtaining the weak one. It applies to the case where two different packets collide with disparate strengths. In Chorus, even two packets with similar strengths can be effectively decoded, because each sees the other as a complement, rather than an interferer.

The PHY layer of Chorus shares similar spirits with the ZigZag protocol [12], which exploits the signal processing capability of SRs to solve the hidden terminal problem in WLANs. ZigZag extracts symbols from collided packets by identifying repeated collisions of two hidden terminals. It treats each collided packet as a sum over two packets. The two original packets are recovered from two known sums, similar to solving a linear system of equations. In the PHY layer, Chorus uses a collision resolution mechanism similar to ZigZag, but it resolves multiple packets from a single collision, given that the packets are identical. In addition, Chorus aims to improve broadcast efficiency in wireless mesh networks, where it exploits transmit diversity and spatial reuse, using MAC-layer cognitive sensing and broadcast scheduling.

Analog network coding (ANC) [13] also exploits PHY layer self-interference cancellation to improve performance of multihop networks. Its rationale is to allow relays one-hop away to transmit concurrently, and cancel out the known packet at the intermediate relay. ANC asymptotically improves relaying throughput, since it reduces the exclusion region from three to two hops, compared to the traditional CSMA. In this paper, we show how ANC can be leveraged to further improve the broadcast performance of Chorus (Section 5.4).

The feasibility of allowing concurrent transmissions to create diversity has also been explored in communications. Concurrent cooperative communication [14], for example, allows colocated wireless nodes to transmit at the same time, thus forming a virtual antenna array that increases signal strength at the common receiver. Beamforming protocols [6] synchronize the transmitters, such that their signals can combine coherently at the receiver. These techniques require strict frequency, phase, and time synchronization at the symbol level, among distributed transmitters. Such fined-grained synchronization remains an open problem [6], due to the limited time resolution at the wireless nodes, and the variation of the wireless channels.

3 Collision Resolution in Chorus

In this section, we introduce the physical-layer collision resolution in Chorus. For clarity, we start with a simple case of two-packet collision, focusing on how to detect, decode, and combine the collided packets to achieve the diversity gain. Then, we deal with the general case of resolving the collision of more than two packets.

3.1 Detecting Collided Packets

In Chorus, a transmitter attaches a known random sequence to the beginning of each packet as a preamble. The receiver then uses a *matched filter* to detect the exact arrival time of this preamble. A matched filter is an optimal linear correlator that maximizes the SNR when correlating unknown signals with a known sequence [15]. It outputs a peak value whenever the packet preamble is detected, even if the preamble is hidden in a strong noise. It operates continuously, so that those preambles overlapping with other packets can still be identified. The number of preambles detected in a run indicates the number of overlapping packets at the receiver.

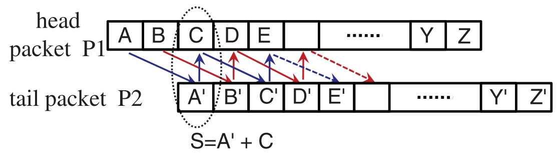


Fig. 2. Iterative decoding of two collided packets carrying the same content.

The peak output grows linearly with the number of bits in the preamble, and with the RSS of the packet [15]. Therefore, the detection threshold is also a linear function of these two factors [12]. In has been observed that using a 32-bit pseudorandom preamble, the collision detection probability is higher than 98 percent under practical wireless settings [12]. Hence, the preamble introduces negligible overhead to the packet.

3.2 Iterative Resolution of Collision

Since a packet usually consists of thousands of symbols, the probability of two collided packets being aligned perfectly is close to 0. In practice, the higher layer operations at transmitters introduce further randomness, resulting in asynchronous arrivals. We identify the natural offset between the two packets by detecting their preambles. Within the offset region, no collision occurs. We first decode the clean symbols therein, and then iteratively subtract such known symbols from the collided ones, thereby obtaining the desired symbol.

For instance, Fig. 2 shows collision of two packets (head packet P1 and tail packet P2) from different transmitters. We first decode the two clean symbols A and B in P1. Symbol C is corrupted as it collides with A' in P2, resulting in a combined symbol S. To recover C, note that symbols A' and A carry the same bit, but the analog forms are different because of channel distortion. Therefore, we need to reconstruct an image of A' by emulating the channel distortion over the corresponding bit that is already known via A. The channel distortion effects, including amplitude attenuation, phase shift, frequency offset, and timing offset, can be accurately estimated using standard communication techniques, as demonstrated in a realistic experimental setting [12].

After reconstruction, we subtract the emulated A' from S, obtaining a decision symbol for C. Then, the decision symbol is normalized using the channel estimation for P1, and a slicer decides if the bit in C is 0 or 1. For BPSK, the slicer outputs 0 if the normalized decision symbol has negative real part, and 1 otherwise. The decoded bit in C is then used to reconstruct C' and decode E. This process iterates until the end of the packet is reached. The iteration for other collided symbols proceeds similarly. The estimation, reconstruction, and cancellation for higher order modulation schemes, such as M-PSK ($M = 4$, 8, 16, 64), can be realized similarly, except that the signal constellation is mapped to different complex numbers [12]. Also, note that the above procedure has linear complexity with respect to packet length, which is similar to ZigZag [12] and interference cancellation [11].

Besides the iterative decoding in the forward direction, Chorus can also work backward, starting from the clean symbols in P2 (i.e., symbol Y' and Z'), to its beginning, hence obtaining a different estimation of the packet. Chorus then performs the following packet combination to improve the decoding probability.

3.3 Using Packet Combination to Improve Diversity

Since P1 and P2 may have different strengths, their decoding confidence also differs. The decoding confidence is indicated by the magnitude of the decision symbol. The farther away it is from the decoding threshold (0 in BPSK), the higher probability it can produce the correct bit, since this is equivalent to a higher SNR. Combining two decision symbols carrying the same bits (e.g., A and A' in Fig. 2) can increase the decoding confidence. This is because the useful information is enhanced, while the noise within the two symbols is not combined coherently.

In Section 5, we show that weighted summing over the corresponding symbols can improve the decoding probability, when two versions of the same packets are received sequentially without collision. Such a weighted combination harvests full transmit diversity, i.e., the SNR of the combined packet is the sum SNR of the two independently received packets.

For those iteratively decoded packets, we only use selective combination, i.e., assigning weight 1 to the packet with the highest SNR, and 0 to all other packets. This is because a weighted combination over two iteratively decoded packets does not improve SNR. In fact, the iterative collision resolution in Chorus can cause error propagation, due to the correlation between consecutively decoded symbols. For example, in Fig. 2, if symbol A produces an erroneous bit, then the error propagates to A', which affects subsequent symbols such as C. Fortunately, such error propagation stops if the actual bits of A' and C are the same. In this case, after subtracting the error image of A', we obtain a strengthened symbol that indicates the correct bit of C. Error propagation also stops when symbol C has a much higher strength than A'. Based on these two observations, we can bound Chorus' BER, showing that the probability of error propagation decays exponentially with the error length (Section 5).

3.4 Multipacket Collision Resolution

Since Chorus allows concurrent transmissions, multiple versions of a packet can collide, especially when the network has high density. The resolution of multipacket collision is complicated by the fact that intermediate packets no longer have clean symbols at the beginning or end. Fig. 3 illustrates a typical scenario.

Let the earliest and latest packets be *head packet* and *tail packet*, respectively. To decode the head packet, Chorus

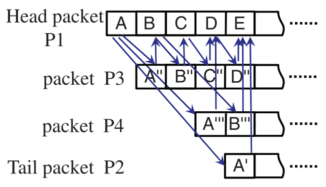


Fig. 3. Collision resolution: The multipacket collision case.

proceeds in a way similar to the two-packet case, except that it needs to subtract multiple reconstructed symbols, including the one from the tail and those from the intermediate packets. Similarly, another version can be obtained by decoding the tail packet, but in reverse order, starting from its end to the beginning. To obtain additional versions from intermediate packets, Chorus performs simple hard decoding. It tracks the packet symbol-by-symbol, treating all others as noise. Intuitively, the results have reasonable confidence only when this packet has a much higher strength than others. The achievable decoding confidence will be rigorously characterized in Section 5.

We recognize that perfect interference cancellation is hard to achieve in practice, due to hardware distortion and imperfect channel estimation. However, the collision resolution mechanism in Chorus does not require perfect interference cancellation. Even though residual interference exists and cannot be completely removed, Chorus is able to decode the useful packet, as long as the SINR is above the decoding threshold. The decoding threshold is around 9.7 dB in typical WiFi receivers [16], whereas even a prototype implementation of interference cancellation can reduce interference by 20 dB [17], i.e., collision resolution can succeed even if interference is 10.3 dB higher than the useful signal. In addition, recent development of specialized hardware and algorithms (e.g., oversampling and long-preamble design [18]) can make the performance closer to ideal interference cancellation.

4 Cognitive Sensing and Broadcast Scheduling

Chorus' physical-layer collision resolution must be integrated with the MAC layer, in order to reduce *unresolvable collisions* occurring when packets with different data collide. In addition, Chorus' network layer must ensure broadcast packets can reach the network edge. Next, we detail both the MAC- and network-layer support for broadcast.

4.1 MAC Layer Cognitive Sensing and Scheduling

Chorus' MAC layer maintains the carrier sensing and backoff in the 802.11-based CSMA protocol, but adopts cognitive sensing that exploits the collision-resolution feature, while avoiding unresolvable collisions. The principle of cognitive sensing is to decode the identity of the packet on the air, and accordingly, make the transmission decision. To this end, Chorus needs to add a new header field into the 802.11 packet.

4.1.1 Chorus Packet Format

Fig. 4 illustrates the broadcast packet format in Chorus. First, a known random sequence is attached to facilitate packet detection and offset identification (Section 3.1).

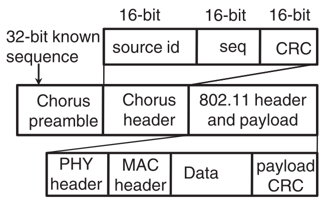


Fig. 4. The broadcast packet format in Chorus.

Second, a *Chorus header* field is added, which informs the receiver of the packet's identity, including the broadcast source's ID and the packet's sequence number. A 16-bit Cyclic Redundancy Check (CRC) [15] is included in this header. In case of CRC failure, this packet is discarded as it conveys wrong identity information.

When the headers of two packets collide, Chorus proceeds with the iterative decoding, assuming they have the same identity. After the decoding, it performs CRC over the header of each packet to ensure they are identical. If not, a decoding failure occurs, and both packets will be discarded. A decoding failure also occurs when the CRC check over the payload fails.

4.1.2 Scheduling of Sensing and Transmissions

With the collision-resolution capability, each transmitter calls a SEND procedure to perform cognitive sensing, as shown in Fig. 5 Transmitters make scheduling decision following three rules:

R1. Forward a packet immediately if the channel is idle.

R2. If the channel is busy, and the packet on the air is exactly one of the packets in the transmit queue, then start transmission of the pending packet.

R3. If the channel is busy, but a preamble cannot be detected, or the header field of the packet on the air cannot be decoded, or a different packet is on the air, then start the backoff procedure according to the 802.11.

R1 is typical of all CSMA protocols. R2 is unique to Chorus's CSMA/CR. It enforces the principle of Chorus, i.e., overlapping packets carrying the same data may not cause collisions. Instead, by collision resolution, these packets offer transmit diversity to the receiver. Therefore, a sender node, such as node B in Fig. 1, can transmit its pending packet if it has the same identity as the one on the air (e.g., the one that A is transmitting). In contrast, CSMA/CA transmitters stall and back off whenever the channel is busy.

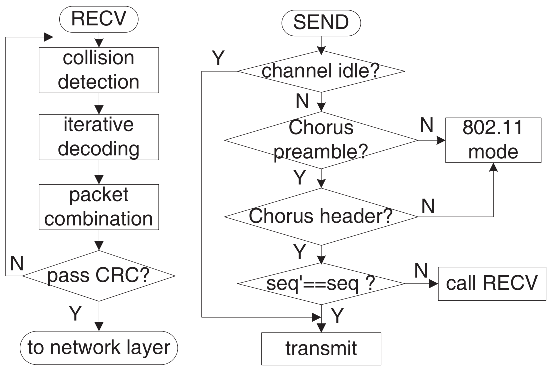


Fig. 5. The MAC-layer control flow in Chorus. seq' denotes the sequence number of the packet on the air.

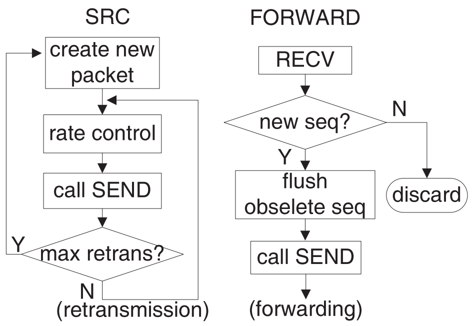


Fig. 6. Control flow for scheduling network-wide broadcast.

R3 ensures friendliness to alien traffic, and is relevant for multisource broadcast and coexistence with CSMA/CA-based unicast traffic. To prevent unresolvable collisions between different packets, Chorus starts the normal 802.11 backoff if it senses that the channel is occupied by such alien traffic. To reduce interference to coexisting traffic, it also backs off conservatively if the identity of the packet on the air cannot be decoded.

The advantages of cognitive sensing and scheduling come at the expense of additional overhead. In 802.11b, the sensing time slot is 20 μs, equivalent to the channel time of 20 bits in the broadcast mode. In contrast, Chorus needs to sense over the entire preamble and the header (80 bits in total, as indicated in Fig. 4). However, this overhead is negligible compared to the packet length (a similar result holds for 802.11a/g/n). We will formalize the cost of the header overhead using both asymptotic analysis (Section 5) and simulation experiments (Section 6).

4.2 Scheduling Network-Wide Broadcast

Compared to existing CSMA/CA-based broadcast protocols, Chorus has the following salient features:

- *Anonymity*. The source and relays do not require any topology information or neighbor identity. As a result, it is insensitive to node mobility, and incurs no control message overhead.
- *Decentralization*. Each node only needs to maintain local states recording the most recent packet ID that it forwarded for a broadcast session (corresponding to a source ID).
- *Diversity*. By decoding multiple copies of the same packet, Chorus gains diversity, and hence, robustness to link losses.

These properties are realized in a wave-propagation style broadcast. Following the SRC procedure in Fig. 6, the source node composes a Chorus packet, and transmits it like a normal 802.11 broadcast packet. *Any* neighbor overhearing this packet will provide best-effort service by forwarding it *once*, following the FORWARD procedure. Receivers with overlapped packets perform collision resolution before continuing with the packet relaying. After each successful reception, a receiver flushes those pending packets with obsolete seq, in order to prevent unresolvable collisions between packets with different sequence numbers. Intuitively, multiple versions of a packet proceed in parallel just like a wavefront, which stops at the network edge.

In case of continuous broadcast of packets, the source distributes a batch of packets (such as software patches) over the network. In such a scenario, Chorus controls the source rate to prevent congestion and avoid collision between packets with consecutive sequence numbers. As verified in the following analysis, the supportable source rate of Chorus allows for a simple closed-form expression that is independent of network topology. This expression can be used to calculate an upper bound on throughput and control the source rate in continuous broadcast.

To improve the reliability of continuous broadcast, the source node can rebroadcast each packet. Intermediate relays need to distinguish rebroadcast packets from the first/original version. This is achieved by splitting the seq field into *broadcast sequence* and packet sequence. To make a tradeoff between delay and reliability, the source node limits the maximum number of retransmissions for each packet. As shown in Section 5.5, a simple relation between retransmission and PDR can be derived for PDR, which can be used as a guideline for making the tradeoff.

When multiple broadcast sessions are running concurrently, their packets are identified through the source-id field in the header part. Each relay maintains a transmit queue storing the packets to be forwarded. When the channel is idle, it directly transmits the head-of-line packet. Otherwise, it follows the MAC-layer cognitive scheduling protocol, which maximizes the spatial reuse opportunity by scheduling the same packets, while avoiding collision with other broadcast sessions. Note that the coexistence with unicast traffic is a special case of multisource broadcast. In effect, the latter case requires more conservative scheduling because of more severe interference, and hence, it will be used as a benchmark for validating Chorus' friendliness to alien traffic.

5 Performance Analysis

In this section, we first characterize the performance of collision resolution and packet combination in Chorus. We then analyze its asymptotic delay and throughput performance, in comparison with the traditional CSMA/CA schemes. The analytical results serve as guidelines for selecting the design parameters, such as packet-combining strategy and maximum source rate.

Unless stated otherwise, we use the following set of notations: L for the packet length, F the offset between two collided packets, D the data rate, W the signal bandwidth, N the noise power, and δ^2 the noise variance. Multiple collided packets are indexed according to their arrival time, and γ_i denotes the SNR of packet i. We maintain consistent settings to the 802.11b broadcast mode. Specifically, all links adopt the 1 Mbps basic access mode using BPSK [9] (assuming $D = 1$ Mbps, $W = 1$ MHz). No MAC-layer retransmission, ACK, RTS/CTS, or other control packets are involved.

5.1 Achievable SNR and BER

We begin with an elementary scenario where two versions of a packet (denoted as P1 and P2) from different transmitters collide. This scenario is analogous to the two-user uplink channel in information theory [5], which adopts interference cancellation as the optimal decoder. However, Chorus' application scenario is unique in that P1 and P2 carry the same content. Ideally, they should complement, or at least do not interfere with each other. This intuition is formalized in the following set of theorems.

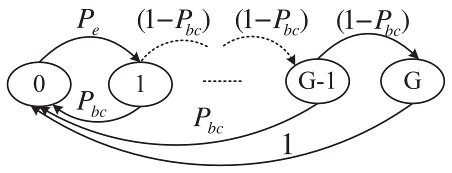


Fig. 7. The error-propagation process as a Markov chain.

Theorem 1. *Without packet combination, the achievable SNR of Chorus' collision resolution in the two-packet collision case is* $\Lambda = \max\{\frac{P_1}{N}, \frac{P_2}{N}\}$. *When decoding* m *overlapped packets, the achievable SNR of Chorus' collision resolution is* $\Lambda = \max\{\frac{P_1}{N}, \frac{P_i}{\sum_{j \neq i} P_j + N}, \frac{P_m}{N}\}$, $i \in \{2, \ldots, m-1\}$.

Proof. The proof follows from Chorus' iterative decoding. We represent symbols in the complex form. Suppose at time t, symbol $\tilde{s}_1(t) = a_1 e^{j\theta_1} x_1(t)$ in P1 collides with $\tilde{s}_2(t) = a_2 e^{j\theta_2} x_2(t)$ in P2. Let v denote the receiver noise, then the received symbol $\tilde{s}(t) = \tilde{s}_1(t) + \tilde{s}_2(t) + v$. If we decode P1 first (forward-direction decoding), then $x_2(t) = x_1(t - F)$. In addition, the channel amplitude a_2 and phase θ_2 can be estimated via correlation, which can achieve high accuracy and introduces negligible noise [12]. Therefore, we can obtain a decision symbol for $x_1(t)$ as: $\tilde{s}(t) - \tilde{s}_2(t) = a_1 e^{j\theta_1} x_1(t) + v$. The resulting SNR level is: $\frac{|a_1 e^{j\theta_1}|^2}{2\delta^2} = \frac{P_1}{N}$, which equals the SNR when $s_1(t)$ is decoded independently.

Similarly, if the clean symbols in P2 are decoded first (backward-direction decoding), then we can obtain $\frac{P_2}{N}$. Taking the maximum of these two yields $\Lambda = \max\{\frac{P_1}{N}, \frac{P_2}{N}\}$.

When m packets collide, the head and tail packets have clean symbols, and the achievable SNRs are $\frac{P_1}{N}$ and $\frac{P_m}{N}$, respectively, following a similar line of reasoning as above. Since Chorus performs hard decoding over intermediate packets, the achievable SNR for an intermediate packet is the same as treating other packets as noise, i.e.,

$$\frac{P_i}{\sum_{j \neq i} P_j + N}, \forall i \in \{2, \ldots, m-1\}.$$

The result follows directly after taking the maximum SNR of all packets. □

The above SNR bounds can be transformed to the BER bound that is directly related to the decoding performance [15]: $\mathrm{BER} = Q(\sqrt{2\Lambda W D^{-1}}) = Q(\sqrt{2\Lambda})$, where the *Q-function* $Q(y) = \frac{1}{\sqrt{2\pi}} \int_y^\infty e^{-\frac{x^2}{2}} \mathrm{d}x$. $Q(y) \to 0$ exponentially when $y < 1$ and $y \to -\infty$, which also holds for $y > 1$ and $y \to \infty$. This implies that BER decreases exponentially with the achievable SNR.

5.2 Effects of Error Propagation

The above SNR and BER bounds are simplified in that they ignore the error propagation along sequentially decoded

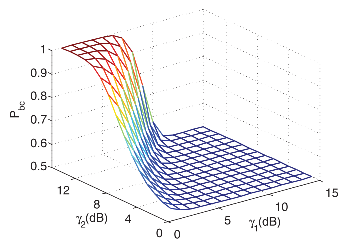


Fig. 8. Head packet's P_{bc}: the probability that error stops propagating to the next bit.

symbols. Fortunately, the following analysis verifies that the error propagation has negligible effect in common cases.

We set up a Markov chain model that relates error propagation to the SNR of each packet, and the offset between collided packets. Again, we start with the two-packet collision scenario in Fig. 2 and analyze the iterative decoding of the head packet P1. As shown in Fig. 7, we define *states* according to the error propagation length, i.e., the number of consecutive errors in a run. The state transition can be classified into two cases: 1) the probability that an independent decoding error occurs (transition from state 0 to state 1), which equals the BER of clean symbols in P1 (denoted as P_e), and 2) the probability P_{bc} that error propagation stops, i.e., the next bit is correct even when the current bit is erroneous. The probability of continuing error propagation is $1-P_{bc}$. The maximum error propagation length starting from a clean symbol is $G=\lfloor\frac{L}{F}\rfloor$, since the distance between any two consecutively decoded symbols equals F.

Obviously, this Markov chain is aperiodic and irreducible, and thus, the steady-state distribution exists. Let π_i be the steady-state probability of state i, then we have the following balance equations:

$$\begin{cases}\pi_1=\pi_0\cdot P_e\\ \pi_i=\pi_{i-1}\cdot(1-P_{bc}), i=2,3,\ldots,G\\ \sum_{i=0}^{G}\pi_i=1.\end{cases}$$

Solving for the steady state, we have

$$\pi_0=\left(1+P_e\cdot(1-(1-P_{bc})^G)P_{bc}^{-1}\right)^{-1}, \tag{1}$$

$$\pi_i=\pi_0\cdot P_e\cdot(1-P_{bc})^{i-1}, i=1,2,\ldots,G. \tag{2}$$

We proceed to derive the probability P_{bc} that error propagation stops. BPSK symbols can be represented as real values subject to channel attenuation, since decoding only depends on the in-phase part of the received symbol. Back to the example in Fig. 2, suppose symbol C carries bit "0" (mapped to -1 in BPSK), and the channel attenuation over C is X_a, then symbol C is represented as $-X_c$. Suppose symbol A' carries bit "1" (mapped to 1 in BPSK) with channel attenuation $X_{a'}$, then the collided symbol $S=-X_c+X_{a'}+v$, where v is the additive white Gaussion noise. In this case, Chorus should subtract $X_{a'}$ from S. However, if the estimation of symbol A is incorrect, it will propagate to C via A'. Specifically, Chorus erroneously subtracts $-X_{a'}$, resulting in a decision value $Y_c=-X_c+2X_{a'}+v$. Similarly, when A' carries bit "0" but Chorus estimates it as "1" via A, the resulting decision value is $Y_c'=-X_c-2X_{a'}+v$. A symmetric argument applies to the case when symbol C carries bit "1." Therefore, the probability that the collision resolution outputs a correct bit is

$$\begin{aligned}P_{bc}&=0.5P\{Y_c'<0\}+0.5P\{Y_c<0\}\\&=0.5P\{w<2X_{a'}+X_c\}+0.5P\{w<X_c-2X_{a'}\}.\end{aligned} \tag{3}$$

The first term in (3) can be bounded as:

$$\begin{aligned}P\{w<2X_{a'}+X_c\}&=1-P\{w\geq 2X_{a'}+X_c\}\\&\geq 1-\delta^2(2X_{a'}+X_c)^{-2}(Chebyshev\ Inequality)\\&=1-(2\sqrt{2\gamma_2}+\sqrt{2\gamma_1})^{-2}.\end{aligned}$$

Both γ_1 and γ_2 are in normal scale, corresponding to practical $\log$ scale values ranging from 6 dB and above [12]. Therefore, in the above equation, it is reasonable to assume $\gamma_1\gg 1, \gamma_2\gg 1$. Consequently, $P\{w<2X_{a'}+X_c\}\approx 1$.

For the second term in (3), a closed-form estimation can be obtained as:

$$\begin{aligned}\Gamma=P\{w<X_c-2X_{a'}\}&=1-\frac{1}{\delta\sqrt{2\pi}}\int_{X_c-2X_{a'}}^{\infty}e^{-\frac{u^2}{2\delta^2}}\mathrm{d}u\\&=1-\frac{1}{\sqrt{2\pi}}\int_{\sqrt{2\gamma_1}-2\sqrt{2\gamma_2}}^{\infty}e^{-\frac{z^2}{2}}\mathrm{d}z\Big(note: z=\frac{u}{\delta}\Big)\\&=1-Q(\sqrt{2\gamma_1}-2\sqrt{2\gamma_2}).\end{aligned}$$

In practice, since the two packets are from two different transmitters, the difference between γ_1 and γ_2 is larger than 1, even in dB scale. Given the exponential decaying of the $Q(\cdot)$ function (Section 5.1), a practical estimation is $\Gamma\approx 1$ if $\gamma_1\gg\gamma_2$ and $\Gamma\approx 0$ if $\gamma_1\ll\gamma_2$.

Combining the analysis of the two terms in (3), we have $0.5\leq P_{bc}\leq 1$, and P_{bc} transits fast from 0.5 to 1 when $\gamma_1\ll\gamma_2$. This trend is also illustrated in Fig. 8.

Back to (1), we have $\pi_0\leq(1+P_e)^{-1}\approx 1-P_e$. π_0 approximates this upper bound as $G\longrightarrow 1$, i.e., the offset between the two packets approaches the packet size. Furthermore, in the common case $G>1$, we have

$$\pi_0\geq\left(1+P_eP_{bc}^{-1}\right)^{-1}\geq(1+2P_e)^{-1}>1-2P_e. \tag{4}$$

Therefore, the bit error probability P_e' in iterative decoding is bounded as:

$$P_e\leq P_e'=1-\pi_0<2P_e. \tag{5}$$

P_e is typically below 10^{-6}; the packet length is around 1KB. Hence, P_e' has a similar effect on the packet error rate (PER) as P_e, even when it approaches the upper bound. This means the *effects of error propagation on PER are negligible*, which will be further verified in our bit-level simulation.

Combining the bounds for P_{bc} and P_e with (2), we conclude that *while resolving a given collision, the error-propagation probability decays exponentially with the error length* (also shown in Fig. 9). This is consistent with the empirical observation in [12]. The above reasoning can easily be extended to multipacket collision resolution, where the probability that error stops propagating is also close to, or larger than 0.5, because the previous erroneous bit may strengthen the current bit with probability 0.5.

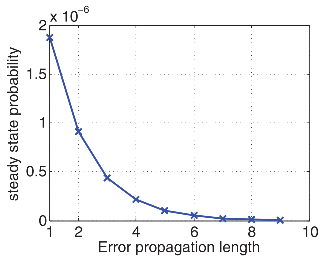


Fig. 9. Steady state distribution of error length. $\gamma_1 = 10, \gamma_2 = 7$. $F = \frac{L}{64}$. Error length 0 is not shown.

5.3 Optimal Packet Combination Weight

When two or more versions of the same packet are received sequentially without any collision, no error propagation occurs. Intuitively, this happens when a small packet size is used. Error propagation is also negligible when the two packets have a large offset F close to packet length. In such cases, we can harvest the transmit diversity via weighted combination of the symbols in the received versions. The optimal weight is derived as follows:

Theorem 2. *Without error propagation, the optimal combination weight of packet i is γ_i. The resulting SNR equals $\sum_{i=1}^{m} \gamma_i$.*

Proof. Suppose m versions of the packets are received sequentially. The tth symbol in version i is $\tilde{s}_i(t) = a_i e^{j\theta_i} x(t) + v$. We normalize the symbol phase, and then assign weight c_i to each version, obtaining a coherently combined decision symbol $\tilde{s}_i(t) = \sum_{i=1}^{m} c_i(a_i x(t) + v)$, resulting in SNR:

$$\widetilde{\mathrm{SNR}} = \frac{\left(\sum_{i=1}^{m} c_i a_i\right)^2}{\sum_{i=1}^{m} c_i^2 \delta^2}. \tag{6}$$

This function can be easily shown to be concave with respect to c_i. Therefore, we take the first-order derivative and obtain

$$c_i = \arg\max_{c_i} \widetilde{\mathrm{SNR}} = \frac{a_i^2}{\delta^2} = \gamma_i. \tag{7}$$

Combining (6) and (7), we obtain: $\max \widetilde{\mathrm{SNR}} = \sum_{i=1}^{m} \gamma_i$, thus completing the proof. □

It should be noted that Theorem 2 does not hold when combining two or more iteratively decoded packets with a small offset F, where the error propagation occurs. In a high SNR region, the error propagation dominates the bit errors caused by noise, so the performance of the weighted combination can be worse than *selective combination*, i.e., assigning weight 1 to the packet with the highest SNR, and 0 to all other packets. This intuition will be further justified via simulation.

5.4 Asymptotic Delay and Throughput

We now analyze the network-level performance of Chorus, including latency and throughput. To be consistent with existing asymptotic analysis [1], [2], [7], we assume perfect reception within the transmission range if no collision occurs. The network radius is r, i.e., it spans r hops from the source to the receiver farthest away.

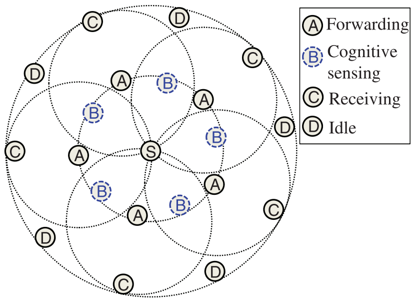


Fig. 10. The worst case latency scenario in Chorus broadcast.

5.4.1 Single-Source Broadcast

We first analyze the case when a single node broadcasts packets over the entire network. Let h denote the size of Chorus preamble plus Chorus header, then we have the following asymptotic performance bound regarding broadcast latency and throughput.

Theorem 3. *The worst case latency and throughput of Chorus is $\frac{r(L+h)}{D}$ and $\frac{LD}{3(L+h)}$*, respectively.

Proof. The network can be divided into r rings centered around the source node. A trivial lower bound on the latency is $r\frac{L}{D}$, i.e., all nodes within the same ring transmit concurrently after those in the previous ring, and the packet's transmission is repeated exactly r times. However, this is achievable only when the cognitive sensing function is disabled. The worst case scenario occurs when cognitive sensing induces the longest delay between adjacent rings, as shown in Fig. 10. Specifically, at most a half of the nodes within each ring transmit while others within the same ring are transmitting. This incurs a latency equal to the duration of the Chorus preamble and header, which equals $\frac{h}{D}$. In addition, the occurrence of this latency can be repeated at most r times over the network, resulting in the worst case latency $r\frac{L+h}{D}$.

In continuous broadcast, packets of different sequences must not collide as such collisions cannot be resolved. To prevent such collisions, nodes within two hops cannot send different packets concurrently. Therefore, a new packet can be sent from the source only after the previous packets have propagated at least three hops, which takes time $3\frac{L+h}{D}$. As a result, the amount of data transmitted within a unit time is $\frac{L}{3\frac{L+h}{D}}$, which is equivalent to the broadcast throughput of Chorus. □

From Theorem 3, one can see that the asymptotic latency of Chorus satisfies $\frac{rL}{D} \le \Theta(r) \le \frac{r(L+h)}{D}$. Under a unit disk graph model, Chorus's latency can be close to the trivial lower bound $\frac{rL}{D}$, since $h \ll L$. This is in sharp contrast with the $\Omega(r \log n)$ latency for anonymous broadcast using CSMA/CA [7].

Theorem 3 also reveals that the maximum supportable source rate (or maximum throughput) of Chorus is insensitive to the network size. As a worst case bound, it can be used to control the source rate in continuous broadcast, in order to prevent the collision between consecutive packets, and avoid congestion.

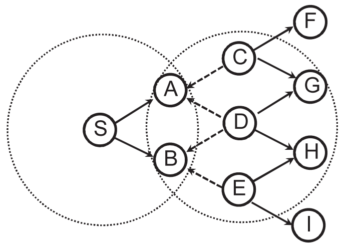


Fig. 11. Snapshot of a continuous broadcast session, where Chorus is combined with analog network coding. Nodes two hops away (S and $\{C, D, E\}$) can broadcast concurrently. Interfering packets from nodes $\{C, D, E\}$ are known to $\{A, B\}$ and can be canceled using analog network coding.

5.4.2 Combining with Analog Network Coding

Chorus exploits self-interference cancellation to resolve collision of packets containing the same data. When the source broadcast a continuous stream of packets, collision may still occur between consecutive packets. By preserving the carrier-sensing mechanism, Chorus alleviates such collision. As a result, the exclusive region (i.e., three hops) is the same as traditional CSMA protocols. However, this problem can be alleviated by combining Chorus with ANC [13].

ANC takes advantage of self-interference cancellation to resolve collision between a known packet and a newly arrived packet. The key observation is that, in multihop wireless networks, a relay node N_r can allow its next-hop and previous-hop nodes to forward packets at the same time. The next-hop node forwards packets known to N_r, which can thus be reemulated and canceled out even when colliding with new packets from the previous-hop node. By combining with ANC, Chorus can allow relays two hops away to forward packets concurrently (as illustrated in Fig. 11), thereby asymptotically improving the throughput performance. Following a similar line of analysis to Theorem 1, we can obtain the following result (we omit the detailed proof).

Corollary 3.1. *When combined with analog network coding, the worst case latency and throughput of Chorus are* $\frac{r(L+h)}{D}$ *and* $\frac{LD}{2(L+h)}$*, respectively.*

In a half-duplex wireless network, any node can either transmit or receive at any time, and thus, the maximum broadcast throughput is bounded by $\frac{D}{2}$. In this sense, Chorus achieves the optimal throughput asymptotically.

In reality, combining Chorus with ANC has more stringent implementation requirements than Chorus itself. To effectively cancel known packets, next-hop and previous-hop relays may need to coordinate with each other, thus complicating the MAC-layer sensing and scheduling mechanisms. Hence, in this paper, we only discuss the theoretical implication of integrating ANC into Chorus, and leave the practical implementation as our future work.

5.4.3 Multiple Broadcast Sessions

We proceed with the case when multiple source nodes broadcast packets concurrently. Following a similar line of analysis with Theorem 3, we can obtain the following result.

Corollary 3.2. *When* M *sessions run Chorus concurrently and the network topology has a bounded node degree* E*, then the worst case latency and throughput of each session are* $\frac{r(L+h)\min\{E^2,M\}}{D}$ *and* $\frac{LD}{(L+h)\min\{E^2,M\}}$*, respectively.*

Proof. For each forwarder of packets of an arbitrary session m, the maximum number of two-hop neighbors is E^2. Thus, the maximum number of contending nodes with alien traffic is $\min\{E^2, M\}$. In the worst case, the packets of session m will be delayed by $\min\{E^2, M\}r$, where r is the network radius (*c.f.*the proof for Theorem 1). Therefore, the worst case delay for a single packet from session m is $\frac{r(L+h)\min\{E^2,M\}}{D}$. Similarly, the throughput bound can be derived as $\frac{LD}{(L+h)\min\{E^2,M\}}$. □

Corollary 3.2 implies that the contention among multiple broadcast sessions may reduce the performance of individual sessions, but does not affect the asymptotic delay and throughput. This is again owing to the wave-front style broadcast enabled by Chorus MAC/PHY layers.

5.5 The Delay-Reliability Tradeoff

When the source node is allowed to rebroadcast each packet (Section 4.2), PDR can be improved, but at the cost of latency. We use the shadowing propagation model [15] to derive a tradeoff between broadcast reliability and latency. The shadowing model accounts for the irregularity of transmission range by modeling the received power at a certain distance as log-normal-distributed. Its accuracy has been verified in outdoor mesh network measurements [19].

With the shadowing model, the received power (in dBW) at a certain distance d is

$$P(d) = P(d_0) - 10\beta \log\left(\frac{d}{d_0}\right) + X, \tag{8}$$

where d_0 is a reference distance with known received power, β the path-loss exponent, X a Gaussian random variable with zero mean and standard deviation σ. $P(d_0)$, β, and σ can be obtained from empirical measurements [19].

Assume the packet reception is successful if the SNR at the receiver side is above a threshold T_s, then the packet-reception probability at distance d is

$$\begin{aligned} P_a(d) &= P\left\{\frac{P(d)}{N} > T_s\right\} = P\left\{X > NT_s + 10\beta\log\left(\frac{d}{d_0}\right)\right\} \\ &= Q\left(\frac{1}{\sigma}\left(NT_s + 10\beta\log\left(\frac{d}{d_0}\right)\right)\right). \end{aligned} \tag{9}$$

Denote R as the network radius in meters. Given a random network, we first partition the topology into c rings centered around the source node with equal-distance ($\frac{R}{c}$) separation. We simplify the analysis by performing a *ring mapping*, which projects each node to the ring closest to it. We consider a degraded version of Chorus where a node can decode at most one packet from transmitters on the previous ring (i.e., the transmit diversity is not exploited). We focus on nodes at the network edge, which are the determinant of delay and PDR. For such nodes, the reception probability is

$$P_c = 1 - \left(1 - P_a^c\left(\frac{R}{c}\right)\right)^{x+1}, \tag{10}$$

where $P_a(\frac{R}{c})$ is the reception probability between nodes on adjacent rings, and x denotes the number of rebroadcast from the source.

On the other hand, the latency of this packet equals the initial delay plus subsequent pipelined retransmissions. When the same packet is retransmitted, three-hops apart nodes can transmit concurrently. In this case, the throughput is $\frac{LD}{3(L+h)}$, following a similarly line of reasoning to Theorem 1. Therefore, the latency of the packet (including retransmission time) is

$$\widetilde{T} = \frac{xL}{\frac{LD}{3(L+h)}} + \frac{c(L+h)}{D} = \frac{L+h}{D}(3x+c). \tag{11}$$

From (10) and (11), we obtain a tradeoff between delay and PDR:

$$P_c = 1 - \left(1 - P_a^c\left(\frac{R}{c}\right)\right)^{\frac{\widetilde{T}D}{3(L+h)} - \frac{c}{3}}. \tag{12}$$

Equation (12) indicates that in a lossy network (corresponding to $P_a^c < 1$), with all other factors fixed, the PDR of Chorus approaches 1 at an exponential speed as delay (or the number of retransmissions) increases. This is especially true for the first few retransmissions. Such a simple relation has not been established for CSMA/CA-based broadcast, because of the intractability of optimal scheduling with CSMA/CA.

6 Experimental Evaluation

We quantitatively evaluate the performance of Chorus in two steps. First, we use symbol-level simulation to verify the effectiveness of its collision-resolution scheme. Then, we introduce the implementation of Chorus based on the 802.11b module in ns-2, and evaluate its broadcast performance in large-scale networks. The simulation experiments further justify our analysis.

6.1 Collision-Resolution Performance

We implement a symbol-level simulator in Matlab. The symbols are represented as complex numbers, whose magnitude depends on the packet's SNR. We assume the receiver noise profile is AWGN, which is a typical approximation to the noise profile after receiver filtering and frequency compensation [12]. Given two or more collided packets, the simulator resolves the collision using Chorus's iterative decoding algorithm. The simulated receiver adopts a simple zero-forcing slicer which outputs a "0" bit if the decision symbol's real part is negative, and "1" otherwise. The signal bandwidth is set to 1 MHz and data rate 1 Mbps. The noise power density is 10^{-11} W/Hz. We vary the received signal power to simulate the SNR range between 0 and 15 dB. For each SNR value, we simulate 5×10^4 collisions, each consisting of three copies of a randomly generated packet of length 1,024 B. The results trivially extend to general cases with an arbitrary number of packets of varying sizes. We focus on the head and tail

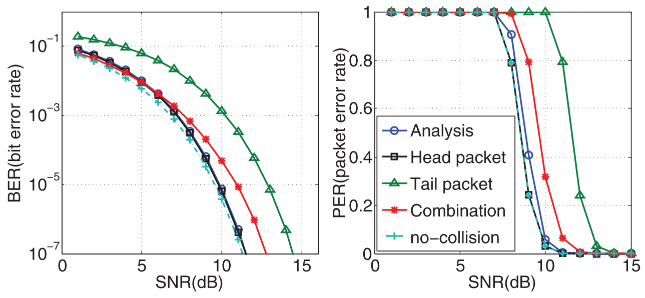


Fig. 12. Symbol-level simulation of iterative decoding. The x-axis represents the SNR of the head packet. As an illustration, the tail packet's SNR is set to 3 dB lower than that of the head packet. *no-collision* indicates the decoding performance when only the head packet is present.

packets since these two adopt iterative decoding while others use hard decoding.

Fig. 12 illustrates the BER and PER of Chorus's iterative decoding algorithm. We observe close performance between Chorus's collision resolution and the case without any collision. This implies that the BER and PER degradation caused by error propagation is negligible under practical settings.

The SNR-weighted combination of decoded packets reduces BER at the low SNR region. However, at the high SNR region, it results in lower performance than selective combination, i.e., assigning weight 1 to the packet with higher SNR, and 0 to the other. This is because as the SNR increases, the error- propagation effect dominates the additional diversity of weighted combination. Our analysis of error propagation (Section 5) is also found to match well with the symbol-level simulation results. Therefore, it can be used as the packet-reception model in the network-level simulation of Chorus.

An additional observation from Fig. 12 is the impact of SNR on Chorus's performance. Inaccurate channel estimation reduces the SNR, thus increasing BER. Our previous analysis assumed accurate channel estimation during the iterative decoding. This is because Chorus detects and decodes collided packets with a relatively high SNR, while treating undetectable packets as noise. In addition, channel estimation is usually realized via adaptive filtering [12], and thus, the noise added is much lower than ambient noise and interference.

6.2 Network-Level Performance

We now evaluate the broadcast performance of Chorus. We implement the cognitive sensing and broadcast scheduling protocols based on the 802.11b module in ns-2 (version 2.33). We adopt the collision-resolution module as the PHY-layer packet-reception model. This module computes the SNR for a given collision pattern, following the analysis in Section 5. The resultant SNR is then compared with the SNR threshold to determine whether the reception succeeds. We do not consider error propagation since it has negligible effect on PER, as shown in our previous analysis and simulation. We only use the selective combination when multipacket collisions occur.

We use a typical CSMA/CA-based protocol, *Double-Coverage Broadcast* [4] as a performance benchmark. In order

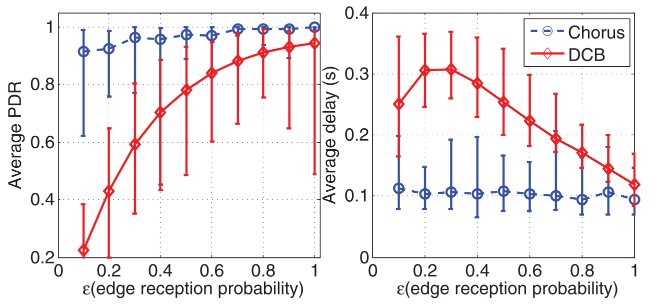


Fig. 13. The impact of link quality (reflected by ϵ) on latency and PDR. The error bars indicate variation over 30 random topologies.

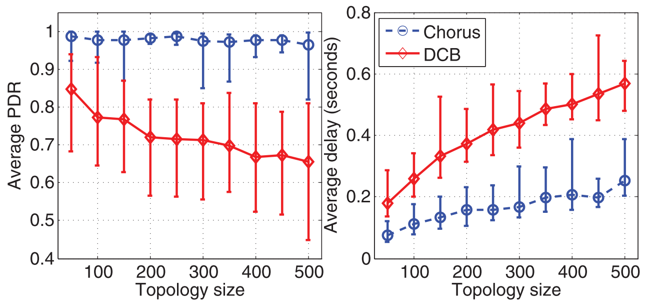


Fig. 14. Scalability of the broadcast protocols as the network size (number of nodes) grows.

to reduce the latency caused by redundant transmissions, DCB prunes the network topology, such that only those nodes with the potential to deliver packets to many downstream receivers will be selected. It further improves PDR by ensuring that each receiver is covered at least twice by other selected forwarders. DCB has been compared with a number of other CSMA/CA-based broadcast protocols and demonstrated its superior performance.

We have implemented DCB based on the ns-2 802.11b MAC, following the specification of [4, Algorithm 5]. Since it requires a strict definition of neighborhood, DCB assumes the existence of a transmission range, within which all nodes receive packets from the transmitter with the same probability. To improve accuracy while satisfying this requirement, we use the following channel model. We define the transmission range at a distance where reception succeeds with an *edge reception probability* ϵ. Within this range, the RSS follows the log-normal distribution [19], with mean 4 and std 5 (dB). This channel model represents a middle ground between the UDG and the log-normal shadowing model. When ϵ is close to 1, it approaches the UDG model. As ϵ approaches 0, it becomes a shadowing model. For a given topology, as ϵ decreases, the average link quality decreases. From the symbol-level simulation in Fig. 12, we observe a sharp decrease of PER beyond certain SNR. Therefore, it is reasonable to assume a SNR threshold exists, above which packets cannot be received. Given the edge reception probability ϵ and noise power, the SNR threshold is calculated by inverting the log-normal function [19].

All experiments are repeated on 30 randomly generated topologies with node degree ranging from 2 to 9. We measure 1) PDR according to the fraction of nodes that successfully receive a packet, and 2) latency equal to the duration between its release and the last successful reception. Both the PDR and latency are averaged over 1,000 packets for each topology, and evaluated with respect to link quality (indicated by ϵ), network size, source rate, and packet size. The typical settings are: source rate 1 pkt/s (packets/second), packet size 1 KB, edge reception probability $\epsilon = 0.5$, network size (number of nodes) 100 with average node density 6. Unless noted otherwise, we isolate the effect of each factor by varying it with others fixed at their typical values.

Our experimental results on DCB are consistent with [4] at high link quality, low source rate, small packet size, and small network size. However, in general, DCB's performance degrades fast. In contrast, Chorus demonstrates significant advantages in all cases. Next, we report the detailed results and their analysis.

6.2.1 Effects of Link Quality

We vary the link quality by tuning the edge reception probability ϵ. A higher ϵ value implies a lower packet loss rate for average links in the network. As shown in Fig. 13, the PDR of both Chorus and DCB decreases with loss rate. However, Chorus is much less sensitive to the link condition, owing to the diversity provided by its collision resolution. As ϵ varies, Chorus's latency remains around 0.1 second, while DCB's latency varies from 0.12 to 0.3. More importantly, Chorus preserves more than 90 percent PDR under all link conditions, while DCB's average PDR drops from 90 to 20 percent as ϵ decreases. Note that DCB's latency may drop as the link quality decreases. This is at the expense of severe packet losses as indicated by the decrease of PDR.

6.2.2 Effects of Network Size

Sensitivity to network size indicates the scalability of broadcast protocol. To quantify the scalability of Chorus, we keep the average network density at 6 while increasing the total number of nodes in the network. The network radius grows accordingly. Fig. 14 plots the resulting latency and PDR. Chorus demonstrates a negligible loss of PDR as the networks size grows. Moreover, its latency is 75 percent lower than that of DCB. Consistent with the asymptotic analysis, its latency increases with the network size. However, the growth rate or sensitivity to network size is much lower than DCB.

6.2.3 Effects of Source Rate

It is well known that in end-to-end unicast or broadcast, the throughput drops when the source rate is too high and the network becomes congested. Therefore, the maximum supportable source rate reflects the maximum throughput of a broadcast protocol. In Fig. 15, we vary the rate at which the source node generates broadcast packets, and track the resulting latency and PDR. Both Chorus and DCB's PDRs decrease abruptly beyond certain values, which roughly indicate their supportable throughput. The supportable throughput of Chorus is around 20 pkts/second, in contrast to 1 pkt/second in DCB. In addition, DCB' latency increases from 0.1 to 10 seconds as the source rate increases from 1 to 40 pkts/second, while Chorus maintains around 0.1 second latency over this range.

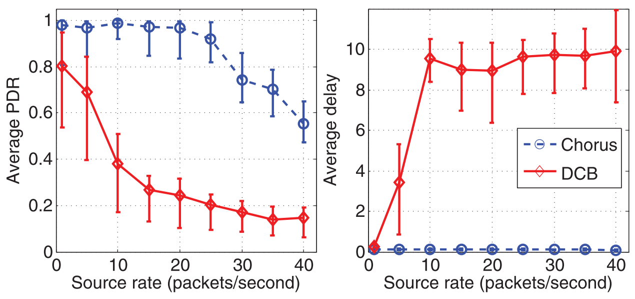


Fig. 15. Sensitivity to the source rate, which indicates the maximum supportable throughput of a broadcast protocol.

6.2.4 Effects of Packet Size

Fig. 16 shows how packet size affects the broadcast performance when coupled with the variation of source rate. When the source rate is low (1 pkt/s), the network is less congested, so Chorus's spatial reuse advantage is less obvious. Owing to the diversity gain, however, it maintains a PDR higher than 95 percent, in contrast with 80 percent when running DCB. Moreover, its latency is 60 percent lower than DCB for all packet sizes. When the source rate is high (10 pkts/s), Chorus's PDR and latency remain the same. In contrast, DCB suffers from a sharp degradation of performance—its latency increases from 0.2 to 4 seconds as packet size grows from 64 to 1,024 B. Again, this is due to its limited supportable throughput. For larger packets, the source injects more data into the network per unit time, which causes congestion. In addition, the cost of losing a packet increases, resulting in higher latency and lower PDR.

As indicated in Section 5, the worst case delay of Chorus is affected by its packet overhead. The experimental results in Fig. 16 show that Chorus is relatively insensitive to packet overhead, in contrast to the analysis. This is because the worst case in Fig. 10 rarely occurs in a random network, and the overhead is negligible compared to packet length.

6.2.5 Effect of Retransmission

Recall the source node can improve the broadcast reliability by rebroadcsting each packet (Section 5.5). To verify this, we generate a number of random topologies with low link quality ($\epsilon = 0.01$, average link quality 0.23). Fig. 17 shows the PDR (averaged over 20 topologies) as the number of retransmissions increases. We see that the PDR increases sharply for the first two retransmissions. However, further retransmissions only make marginal improvement, because randomly generated topologies tend to contain a number of isolated nodes with sparse connections to the majority of nodes. The PDR of such nodes remains low even with retransmissions, especially in extremely lossy networks.

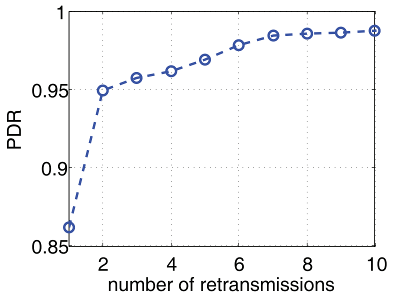


Fig. 17. Effect of source retransmission on the reliability of broadcast.

6.2.6 Broadcast in Mobile Networks

Traditional broadcast protocols is often hampered by node mobility, since they need timely topology updates in order to recalculate the optimal set of forwarders. However, a topology update often involves network-wide broadcast for neighbor discovery and other control overhead. Chorus removes this obstacle since it enables anonymous broadcast forwarding (Section 4.2).

Fig. 18 verifies this intuition by running Chorus at various node speeds (the number of nodes is 200 and $\epsilon = 0.5$). We compare Chorus with an ideal version of DCB, where all nodes can obtain timely update of the topology information and forwarder set assignment from an oracle. We see that for both protocols, the PDR and delay are insensitive to node mobility. In addition, Chorus still maintains high PDR and low latency compared to the ideal DCB. Its performance gain is unaffected by node mobility.

6.2.7 Effects of Network Density

Network density, indicated by the average node degree (number of neighbors), affects the amount of redundancy or diversity that can be harvested from repeated transmissions of the same packets. Fig. 19 quantifies this effect. We fix the node population at 200 and ϵ at 0.5. The network decreases as network density increases, resulting in a higher PDR. When the density exceeds 15, DCB achieves a similar level of PDR as Chorus. However, such a high density implies that all nodes are clustered in a small collision domain, which lacks generality in practice. In addition, even under a high node density, Chorus can still achieve more than 53 percent delay reduction compared to DCB.

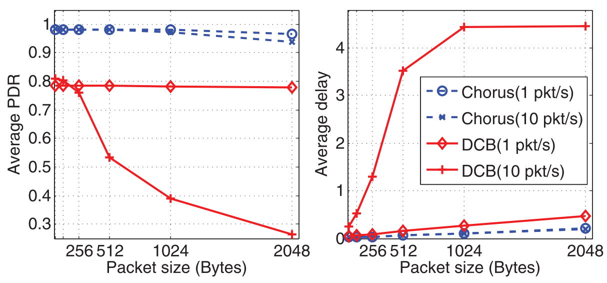


Fig. 16. Impact of packet size, ranging from 64 to 2,048 bytes.

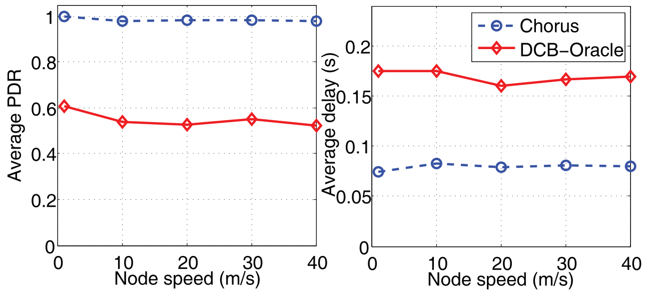


Fig. 18. Effects of node mobility on broadcast performance. DCB-Oracle is an ideal DCB protocol that tracks the topology without any message overhead.

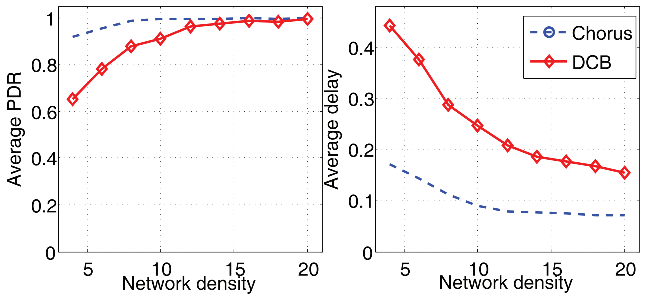


Fig. 19. Effect of network density (reflected by average node degree) on latency and PDR. Note that the number of nodes is kept at 200.

6.2.8 Multiple Broadcast Sessions

We proceed to evaluate the case where multiple broadcast sessions coexist, each corresponding to one randomly selected source node in a 50-node topology. We set $\epsilon = 0.1$ and $\epsilon = 0.5$ to represent lossy and nonlossy networks, respectively. The former is close to a real-world mesh network [20] in which most links have an intermediate reception rate. We focus on two metrics: average PDR among all sessions, and broadcast throughput, which equals the total amount of data delivered to all nodes within unit time, summed over all the sessions. Fig. 20 plots these metrics as a function of traffic load (the number of sessions). In a lossy network, Chorus achieves a $3\times$ higher throughput than DCB, and maintains PDR above 60 percent, which indicates the friendliness among different traffic. The performance gain over DCB is less in a nonlossy network, where Chorus benefits more from spatial reuse than diversity gain. Also, although throughput increases when the traffic load is high, the cost is lower PDR, implying that most traffic is confined to the vicinity of source nodes, especially for the DCB protocol.

7 Conclusion

In this paper, we provide theoretical and practical results that demonstrate the feasibility and advantages of a collision-resolution protocol for wireless broadcast. We introduce Chorus, which allows forwarders with the same outgoing packets to transmit roughly at the same time, and then employs physical-layer iterative decoding to resolve collisions at the receiver. By decoding multiple versions of a packet at once, Chorus achieves transmit diversity and improves loss resilience without any retransmission. More importantly, with its collision-tolerant MAC, Chorus significantly simplifies the CSMA scheduling and improves its spatial reuse. Our theoretical analysis and symbol-level simulation show that Chorus's iterative decoding algorithm can effectively resolve collisions with negligible error propagation effect. We also establish an asymptotic latency bound of $\Theta(r)$ when using Chorus for broadcast, where r is the network radius. Our network-level simulation experiments further show that Chorus outperforms a typical CSMA/CA-based broadcast protocol by a significant margin, in terms of latency, reliability, throughput, and scalability. These features make Chorus suitable especially for fast information dissemination in large-scale networks, such as wireless mesh networks.

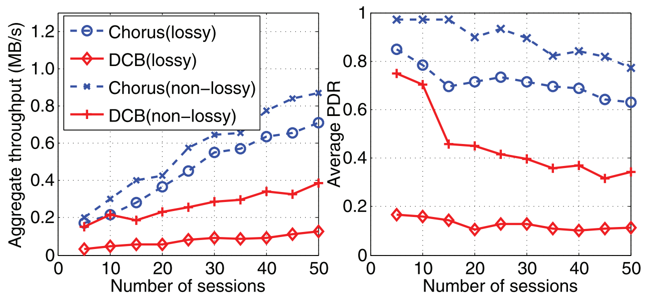


Fig. 20. Total broadcast throughput and average PDR when multiple sources transmit different data, for lossy (edge reception probability $\epsilon = 0.1$, average link quality $q = 0.51$) and nonlossy ($\epsilon = 0.5, q = 0.83$) networks.

Acknowledgments

The work reported in this paper was supported in part by the US National Science Foundation under grants CNS-0721529 and CNS-1114837.

References

[1] R. Gandhi, S. Parthasarathy, and A. Mishra, "Minimizing Broadcast Latency and Redundancy in Ad Hoc Networks," *Proc. ACM MobiHoc*, 2003.

[2] S.-H. Huang, P.-J. Wan, X. Jia, H. Du, and W. Shang, "Minimum-Latency Broadcast Scheduling in Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM*, 2007.

[3] S. Huang, P.J. Wan, J. Deng, and Y. Han, "Broadcast Scheduling in Interference Environment," *IEEE Trans. Mobile Computing*, vol. 7, no. 11, pp. 1338-1348, Nov. 2008.

[4] W. Lou and J. Wu, "Toward Broadcast Reliability in Mobile Ad Hoc Networks with Double Coverage," *IEEE Trans. Mobile Computing*, vol. 6, no. 2, pp. 148-163, Feb. 2007.

[5] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication.* Cambridge Univ., 2005.

[6] R. Mudumbai, D.R. Brown, U. Madhow, and H.V. Poor, "Distributed Transmit Beamforming: Challenges and Recent Progress," *IEEE Comm. Magazine*, vol. 47, no. 2, pp. 102-110, Feb. 2009.

[7] B.S. Chlebus, L. Gasieniec, A. Gibbons, A. Pelc, and W. Rytter, "Deterministic Broadcasting in Unknown Radio Networks," *Proc. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2000.

[8] R. Mahjourian, F. Chen, R. Tiwari, M. Thai, H. Zhai, and Y. Fang, "An Approximation Algorithm for Conflict-Aware Broadcast Scheduling in Wireless Ad Hoc Networks," *Proc. ACM MobiCom*, 2008.

[9] *IEEE 802.11 Standard: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE, 2007.

[10] C. Fragouli, J. Widmer, and L.B. Jean-Yves, "Efficient Broadcasting Using Network Coding," *IEEE/ACM Trans. Networking*, vol. 16, no. 2, pp. 450-463, Apr. 2008.

[11] D. Halperin, T. Anderson, and D. Wetherall, "Taking the Sting Out of Carrier Sense: Interference Cancellation for Wireless LANs," *Proc. ACM MobiCom*, 2008.

[12] S. Gollakota and D. Katabi, "ZigZag Decoding: Combating Hidden Terminals in Wireless Networks," *Proc. ACM SIGCOMM*, 2008.

[13] S. Katti, S. Gollakota, and D. Katabi, "Embracing Wireless Interference: Analog Network Coding," *Proc. ACM SIGCOMM*, 2007.

[14] A. Scaglione and Y.-W. Hong, "Opportunistic Large Arrays: Cooperative Transmission in Wireless Multihop Ad Hoc Networks to Reach Far Distances," *IEEE Trans. Signal Processing*, vol. 51, no. 8, pp. 2082-2092, Aug. 2003.

[15] B. Sklar, *Digital Communications: Fundamentals and Applications.* Prentice Hall, 2001.

[16] B. McFarland, A. Shor, and A. Tabatabaei, "A 2.4 & 5 GHz Dual Band 802.11 WLAN Supporting Data Rates to 108 Mb/s," *Proc. Technical Digest Gallium Arsenide Integrated Circuit Symp.*, 2002.

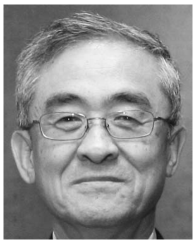
[17] J.I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti, "Achieving Single Channel, Full Duplex Wireless Communication," *Proc. ACM MobiCom*, 2010.

[18] J. Yang and R.W. Brodersen, "Time Domain Interference Cancellation for Cognitive Radios and Future Wireless Systems," PhD thesis, Electrical Engineering and Computer Science Dept., Univ. of California, Berkeley, May 2010.

[19] J. Camp, J. Robinson, C. Steger, and E. Knightly, "Measurement Driven Deployment of a Two-Tier Urban Mesh Access Network," *Proc. ACM MobiSys,* 2006.

[20] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and Evaluation of an Unplanned 802.11b Mesh Network," *Proc. ACM MobiCom,* 2005.

Xinyu Zhang received the BEng degree in 2005 from the Harbin Institute of Technology, China, and the MS degree in 2007 from the University of Toronto, Canada. He is currently working toward the PhD degree in the Department of Electrical Engineering and Computer Science, University of Michigan. His research interests are in the MAC/PHY codesign of wireless networks, with applications in WLANs, WPANs, mesh networks, and white-space networks. He was the recipient of the Best Paper Award at ACM MobiCom 2011.



Kang G. Shin is the Kevin & Nancy O'Connor professor of computer science in the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor. His current research focuses on computing systems and networks as well as on embedded real-time and cyber-physical systems, all with emphasis on timeliness, security, and dependability. He has supervised the completion of 69 PhDs, and authored/coauthored more than 770 technical articles (more than 270 of these are in archival journals), one textbook, and more than 20 patents or invention disclosures. He has received numerous awards, including Best Paper Awards from the 2011 ACM International Conference on Mobile Computing and Networking (MobiCom 2011), the 2011 IEEE International Conference on Autonomic Computing, the 2010 and 2000 USENIX Annual Technical Conferences, as well as the 2003 IEEE Communications Society William R. Bennett Prize Paper Award and the 1987 Outstanding IEEE Transactions of Automatic Control Paper Award. He has also received several institutional awards, including the Research Excellence Award in 1989, the Outstanding Achievement Award in 1999, the Distinguished Faculty Achievement Award in 2001, and the Stephen Attwood Award in 2004 from The University of Michigan (the highest honor bestowed to Michigan Engineering faculty); a Distinguished Alumni Award of the College of Engineering, Seoul National University in 2002; the 2003 IEEE RTC Technical Achievement Award; and the 2006 Ho-Am Prize in Engineering (the highest honor bestowed to Korean-origin engineers). He has chaired several major conferences, including ACM MobiCom 2009, IEEE SECON 2008, ACM/USENIX MobiSys 2005, IEEE RTAS 2000, and IEEE RTSS 1987. He has served on editorial boards including the *IEEE Transactions on Parallel and Distributed Systems* and the *ACM Transactions on Embedded Systems*. He has also served or is serving on numerous government committees, such as the US National Science Foundation Cyber-Physical Systems Executive Committee and the Korean Government R&D Strategy Advisory Committee. He has also cofounded a couple of startups. He is a fellow of the IEEE and the ACM.