PPR: Partial Packet Recovery for Wireless Networks

Kyle Jamieson and Hari Balakrishnan

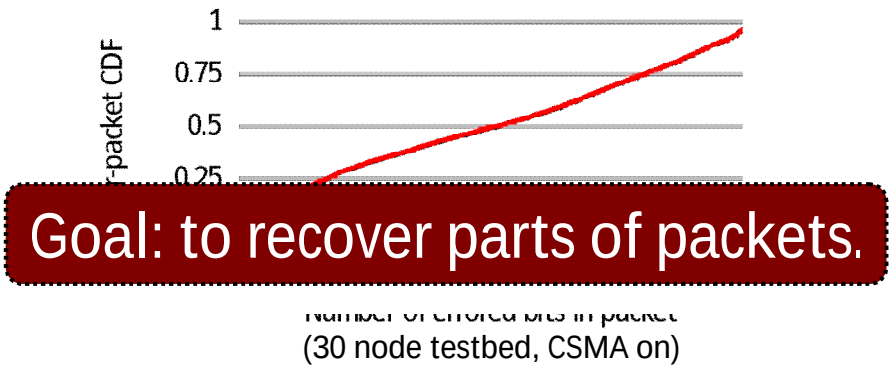
MIT Computer Science and Artificial Intelligence Laboratory

The problem

- Lots of packets lost to collisions and noise in wireless networks

Can't recover non-colliding bits today!

Bits in a packet don't share fate



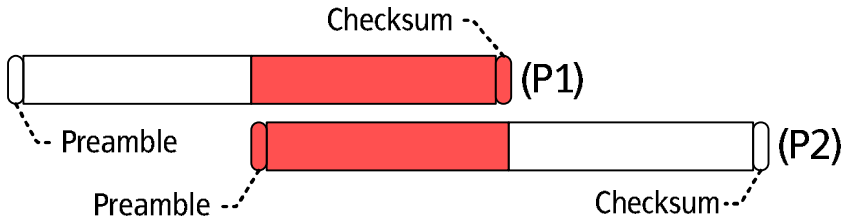
Goal: to recover parts of packets.

(30 node testbed, CSMA on)

Many bits from corrupted packets are correct, but status quo receivers don't know which!

3

Three key questions



1. How does receiver know which bits are correct?
2. How does receiver know P2 is there at all?
3. How to design an efficient ARQ protocol?

4

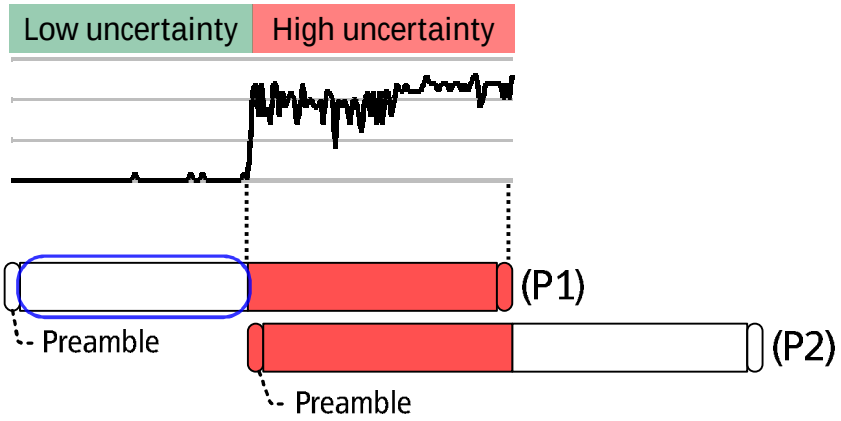
How can receiver identify correct bits?

- Use physical layer (PHY) hints: SoftPHY
  - Receiver PHY has the information!
  - Pass this confidence information to higher layer as a hint
- SoftPHY implementation is PHY-specific; interface is PHY-independent
- Implemented for direct sequence spread spectrum (DSSS) over MSK (this talk) and other modulations

5

A new source of information

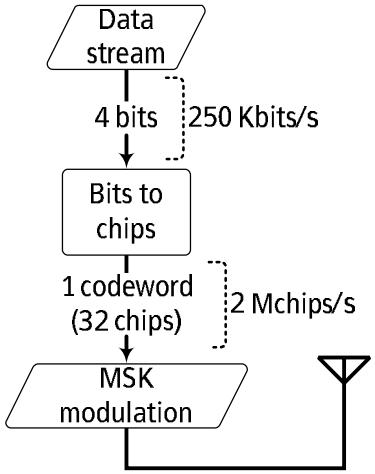
PHY conveys uncertainty in each bit it delivers up



6

Direct seq. spread spectrum background

Transmitter:



Receiver:

- Demodulate MSK signal
- Decide on closest codeword to received (Hamming distance)
- Many 32-bit chip sequences are not valid codewords
- Codewords separated by at least 11 in Hamming distance
- 802.11 similar

7

SoftPHY hint for spread spectrum

Hamming distance between received chips and decided-upon codeword

Receive: **11101101000111000011010110100010**
C_1: **11101101100111000011010100100010**
⇒ SoftPHY hint is 2

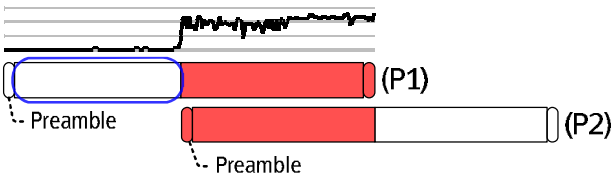
Receive: **11001101000111010111011110110111**
C_1: **11101101100111000011010100100010**
⇒ SoftPHY hint is 9

8

Three key questions

1. How does receiver know which bits are correct?

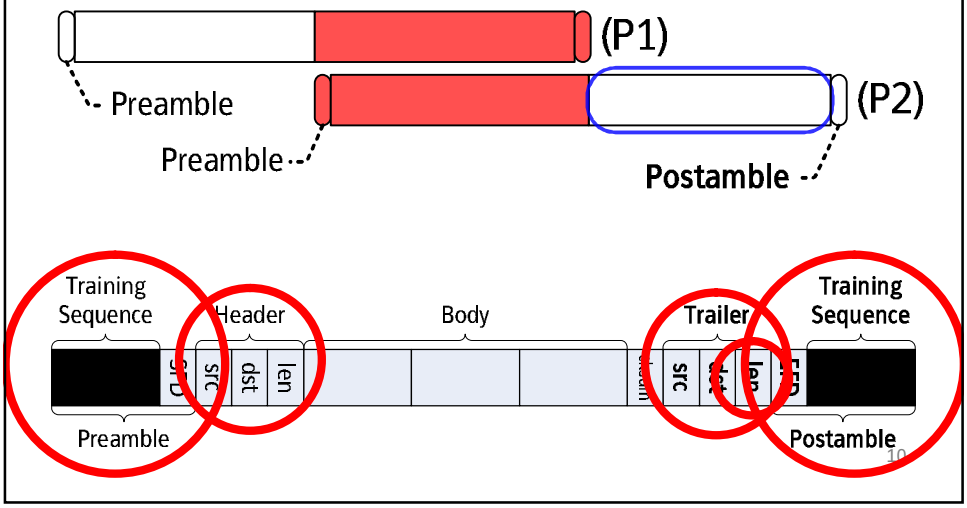
   A: SoftPHY:



2. How does receiver know P2 is there at all?
3. How to design an efficient ARQ protocol?

9

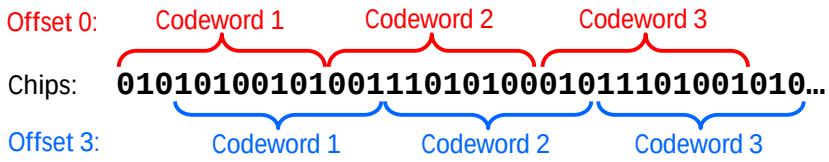
Postamble decoding



10

Receiver design with postamble

- Codeword synchronization
  - Translate stream of chips to codewords
  - Search for postamble at all chip offsets



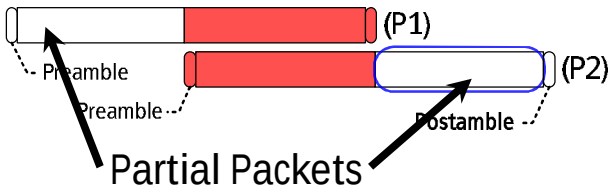
- Chip synchronization without preamble/postamble

11

Three key questions

1. How does receiver know which bits are correct?
2. How does receiver know P2 is there at all?

A: Postamble:

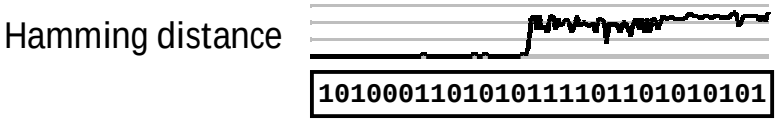


3. How to design an efficient ARQ protocol?

12

ARQ with partial packets

- ARQ today: correctly-received bits get resent
- PP-ARQ key idea: resend only incorrect bits

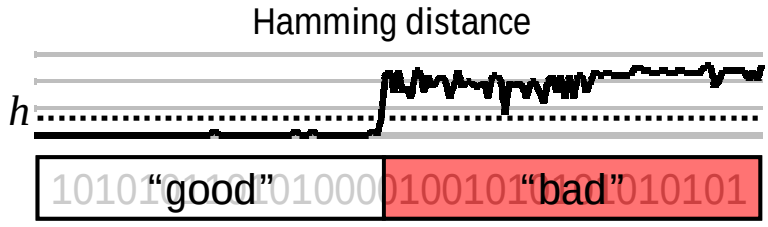


- Efficiently tell sender about what happened
  - Feedback packet

13

Labeling bits “good” or “bad”

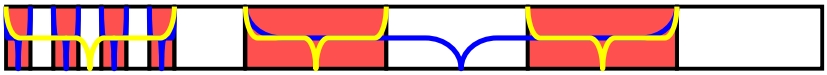
- Threshold test: pick a threshold h
  - Label codewords with SoftPHY hint > h “bad”
  - Label codewords with SoftPHY hint $\leq h$ “good”



14

PP-ARQ protocol

1. Assuming hints correct, which ranges to ask for?
   - Dynamic programming problem
   - Forward and feedback channels



2. Codewords are in fact correct or incorrect
   - Two possibilities for mistakes
     - Labeling a correct codeword “bad”
     - Labeling an incorrect codeword “good”

“Good” bits
“Bad” bits

15

Implementation

Sender: telos tmote sky sensor node
- Radio: CC2420 DSSS/MSK (Zigbee)
- Modified to send postambles

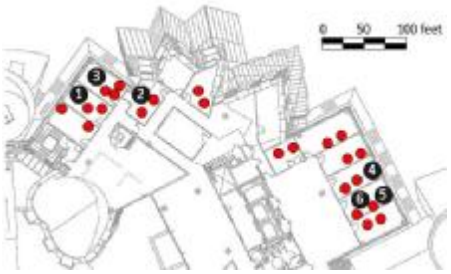
Receiver: USRP software radio with 2.4 GHz RFX 2400 daughterboard
- Despreading, postamble synchronization, demodulation
- SoftPHY implementation

PP-ARQ: trace-driven simulation using data from above

16

Experimental design

- Live wireless testbed experiments
  - Senders transmit 101-byte packets, varying traffic rate
  - Evaluate raw PPR throughput
  - Evaluate SoftPHY and postamble improvements



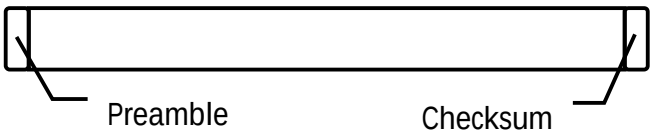
- Trace-driven experiments
  - Evaluate end-to-end PP-ARQ performance
  - Internet packet size distribution
  - 802.11-size preambles

25 senders
6 receivers

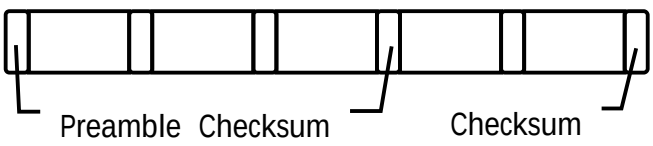
17

PP-ARQ performance comparison

- Packet CRC (no postamble)

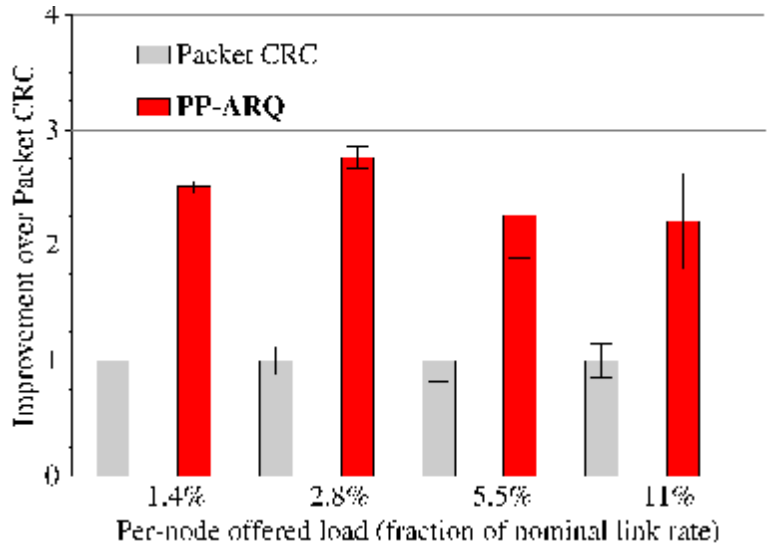


- Fragmented CRC (no postamble)
  - Tuned against traces for optimal fragment size

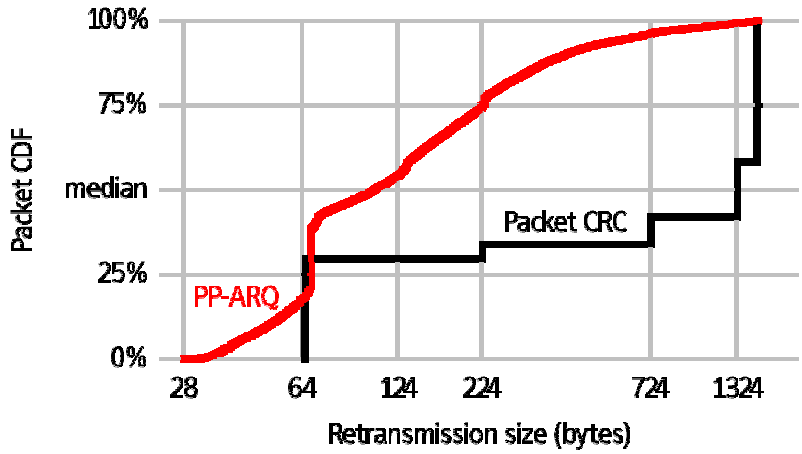


18

Throughput improvement 2.3-2.8x

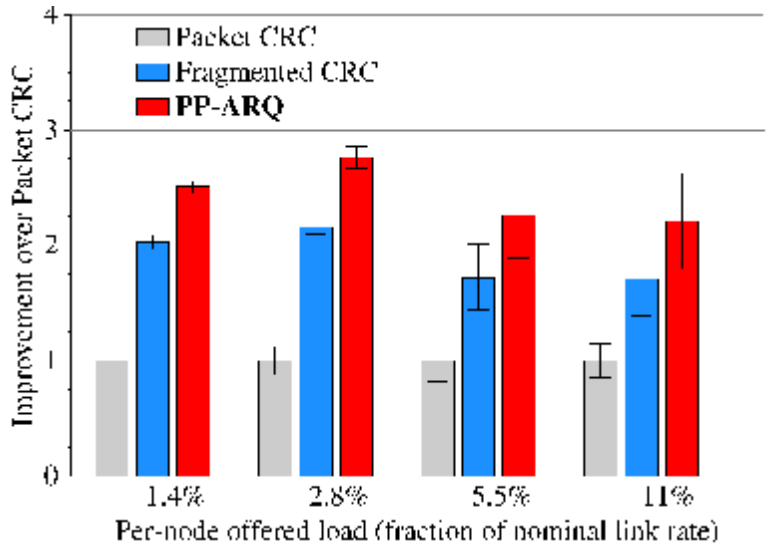


PP-ARQ retransmissions are short

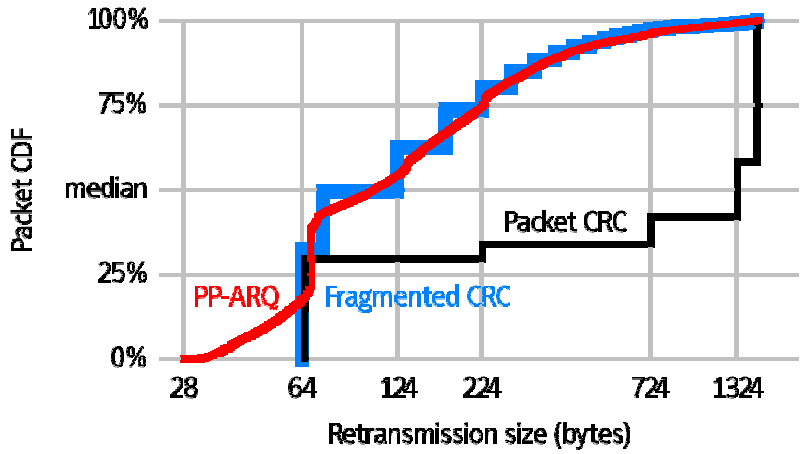


20

25% improvement over fragmented

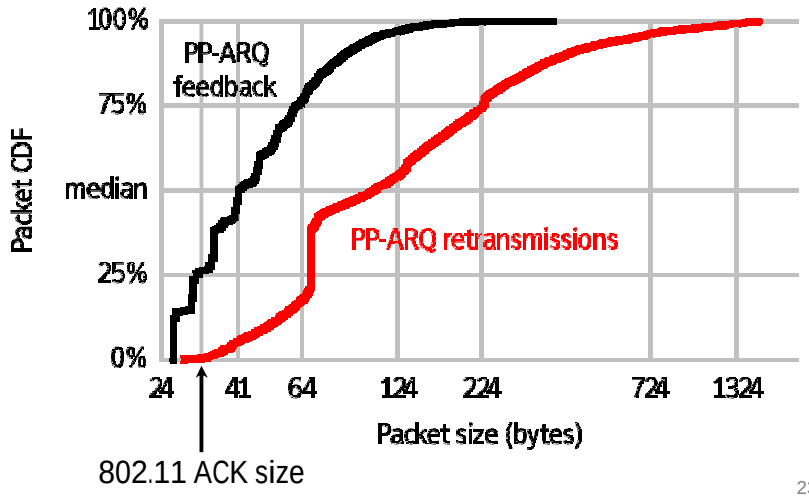


PP-ARQ retransmissions are short



22

PP-ARQ feedback overhead is low



23

Related work

- ARQ with memory [Sindhu, IEEE Trans. On Comm. '77]
  - Incremental redundancy [Metzner, IEEE Trans. On Comm. '79]
  - Code combining [Chase, IEEE Trans. On Comm. '85]
- Combining retransmissions
  - SPaC [Dubois-Ferrière, Estrin, Vetterli; SenSys '05]
- Diversity combining
  - Reliability exchanging [Avudainayagam et al., IEEE WCNC '03]
  - MRD [Miu, Balakrishnan, Koksal; MobiCom '05]
  - SOFT [Woo et al.; MobiCom '07]
- Fragmented CRC
  - Seda [Ganti et al.; SenSys '06], 802.11 fragmentation

24

Conclusion

- Mechanisms for recovering correct bits from parts of packets
  - SoftPHY interface (PHY-independent)
  - Postamble decoding
- PP-ARQ improves throughput 2.3–2.8× over the status quo
- PPR Useful in other apps, e.g. opportunistic forwarding

25