Rateless Deluge: Over-the-Air Programming of Wireless Sensor Networks using Random Linear Codes

IPSN 2008

Presented by L. C. Yang
2008/5/29
Outline

- Introduction
- Related Work
- Proposed approach
  - Rateless Deluge
  - ACKless Deluge
- Experiment and Simulation
- Conclusion
Introduction

- Sensor networks’ software often needs to be updated after deployment.

- Over-the-air programming (OAP) protocols play a key role as an enabling technology to numerous sensor network applications.
Introduction

- Performance **degrades** when the network **size** and **density** get **large**, and when packet **loss is high**.

- This **lack scalability** can largely be attributed to the **high control overhead**, most specifically with (NACK) mechanisms.

- **NACK implosion problem**.
This paper use rateless coding to eliminate the need to convey control information about which packets require retransmission.
Deluge

- Divides data image into pages, each page consisting of a fixed number of packets.

- Piplining.

- NACK-based protocol.
Deluge

Advertisement

Request

Transmit data

Request

Advertisement

1 0 0 0 0 0

1 1 1 1 1 1

0 0 0 0 0 0
Pipelining
Rateless and Random Linear Codes

- Do not need to indicate which packets require retransmission; just have to receive a sufficient number of packets can be used to decode.

- Communication and energy savings.

- Lower control overhead.
Rateless and Random Linear Codes

\[ F \supset \begin{cases} X_1 \\ X_2 \\ X_3 \\ X_K \end{cases} \times \beta_{i,1} \times \beta_{i,K} \]

original file

encoded file

\[ \begin{aligned} Y_1 \\
Y_2 \\
Y_3 \\
Y_i \\
Y_n \end{aligned} \]
Rateless and Random Linear Codes

Traditional:
Request Lost Packets:

Rateless Codes:
Request Lost Packets:
Rateless Deluge

- Modify the original Deluge protocol in that it uses rateless codes to transmit data.

- This change causes the communication in the control and data planes are reduced.
Rateless Deluge

- Change for request mechanism:
  - Only need the number of missing packet instead of bit vector.

- Change for sender:
  - Encoding before send (cost time).
  - Precoding.

- Change for receiver:
  - Decoding after receive.
State diagram for sender

1. Request
2. Load(data)
   - if all loaded
     - Encode()
   - else
     - Load(next)
3. Proc(page)
   - if valid
     - if loaded
       - Encode()
     - else
       - Load(first)
4. Precoder()
   - if next
     - Proc(next)
   - else
     - Done
5. Encode()
   - if sent all
     - Precoder()
   - else
     - Encode()
6. Done
State diagram for receiver

Data Packet
- Receive()
  - num++
  - if num = n
    - Decode()

- Decode()
  - if correct
    - Write(data)
  - else
    - Discard()

- Discard()
  - for all data
    - if dependent
      - delete
  - Wait

- Write(data)
  - if all written done
    - else
      - Write(next)

- Wait
- Done
ACKless Deluge

- Attempts to completely eliminate the need for NACKs.
  - no waiting time for precoding

- Employs a FEC mechanism at the packet level which sends extra encoded packets in addition to the requested number of packets.

- FEC is dependent on number of neighbor and probability of loss.
Experiment

- 1-Hop 16 motes
- 9-pages image 20-packets per page
Experiment

- N/2 1-Hop
- N/2 2-Hop
- 9-pages image

![Graph showing time to disseminate vs number of motes]

- Rateless Deluge
- ACKless Deluge
- Original Deluge
Simulation

9-pages image

7% Packet loss rate

![Graph showing network data packets and energy consumption vs. number of nodes for different Deluge variants.]
Conclusion

- This paper shows the benefits of using random linear codes for over-the-air programming of sensor networks.

- We expect that rateless code transfer mechanisms are practical and useful for any communication protocol in wireless sensor networks.