PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks

Chieh-Yih Wan, Andrew Campbell,
Lakshman Krishnamurthy

WSNA'02 (JSAC'05)

Adopted from the in-class presentation by Thanos Stathopoulos at UCLA

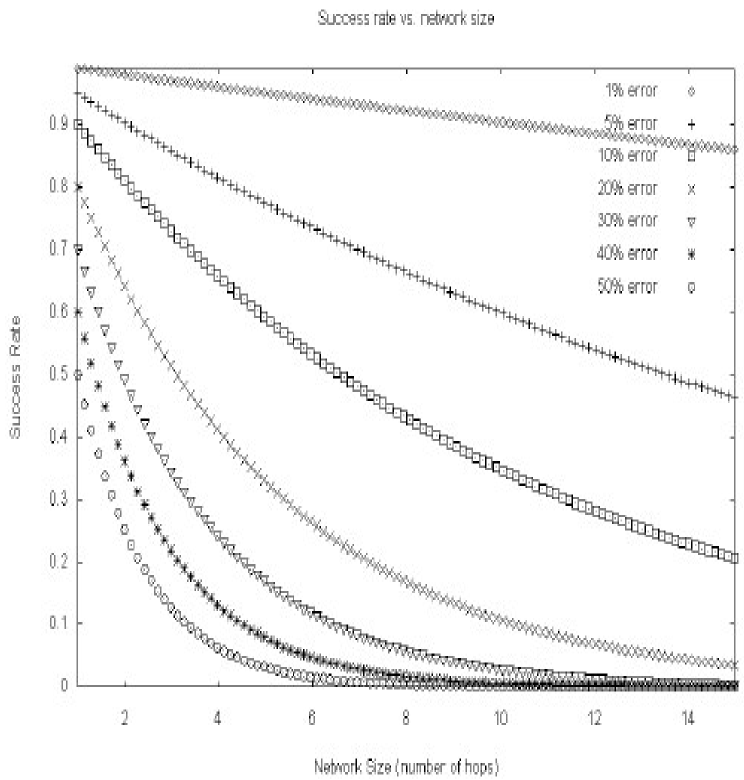
Motivation

- Most sensor network applications do not need reliability?
  - Sources => sink.
- New applications like re-tasking of sensors need reliable transport.
  - Sink => sources.
- Current sensor networks are application specific and optimized for that purpose.
- Future sensor networks may be general purpose to some extent – ability to re-program functionality.

Design Goals of Reliable Transport Protocol in WSN

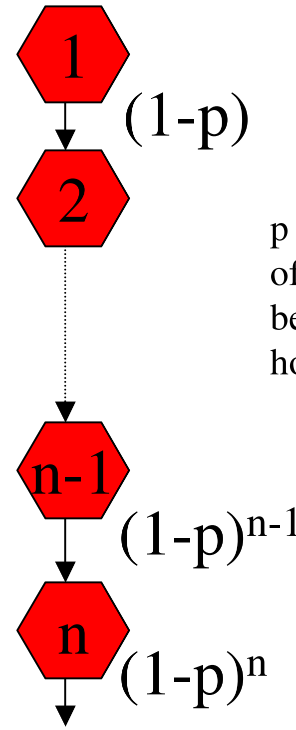
- Simplicity.
- Robustness.
- Scalability.
- Customizability.

End-to-End Considered Harmful



$(1-p)^n$

Prob. to detect loss in NACK system.



p is the error rate of wireless link between two hops

❒Probability of reception degrades exponentially over multiple hops

Hop-by-Hop Error Recovery

- Intermediate nodes now responsible for error detection and recovery
  - Loss detection probability is now constant
    - Exponential decrease in end-to-end
- Cost: Keeping state on each node
  - Potentially not as bad as it sounds!
    - Cluster/group based communication
    - Intermediates are usually receivers as well

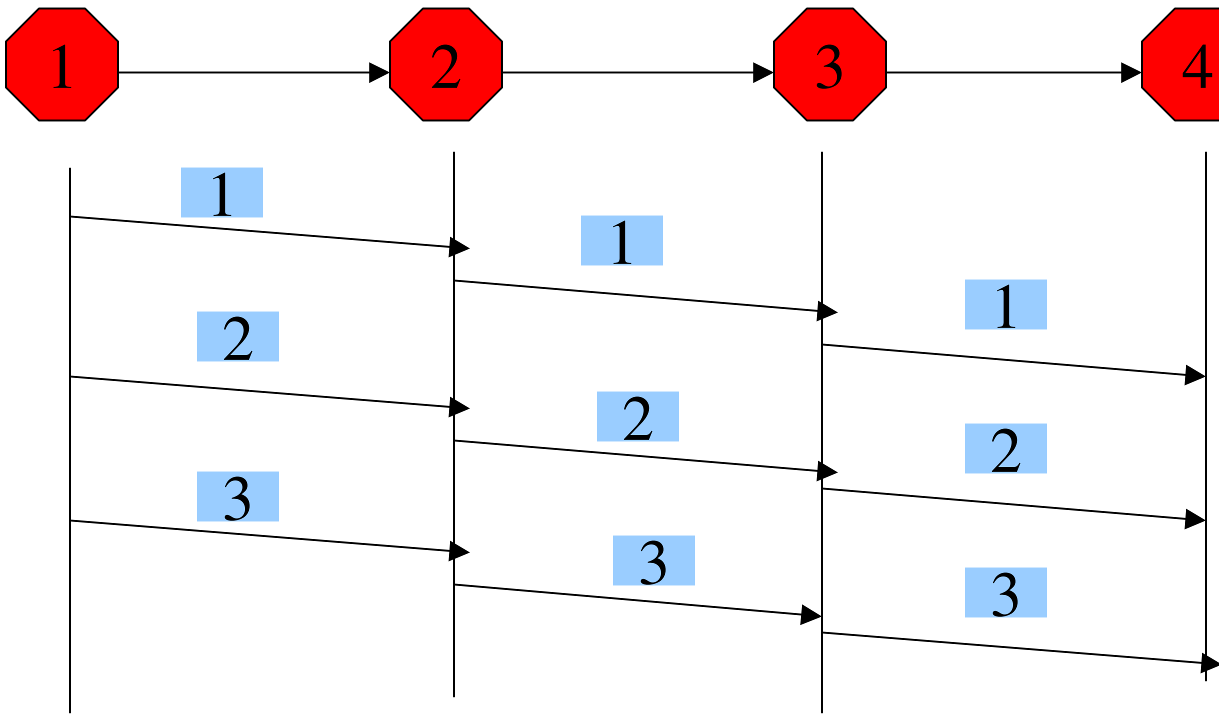
Pump Slowly, Fetch Quickly (PFSQ)

- Slow data distribution (pump slowly)
- Quick error recovery (fetch quickly)
  - Assumption: no congestion, losses due only to poor link quality
- Goals
  - Recover from losses locally.
  - Ensure data delivery with minimum support from transport infrastructure
  - Minimize signaling overhead for detection/recovery operations
  - Operate correctly in poor link quality environments
  - Provide loose delay bounds for data delivery to all intended receivers

PSFQ Operation

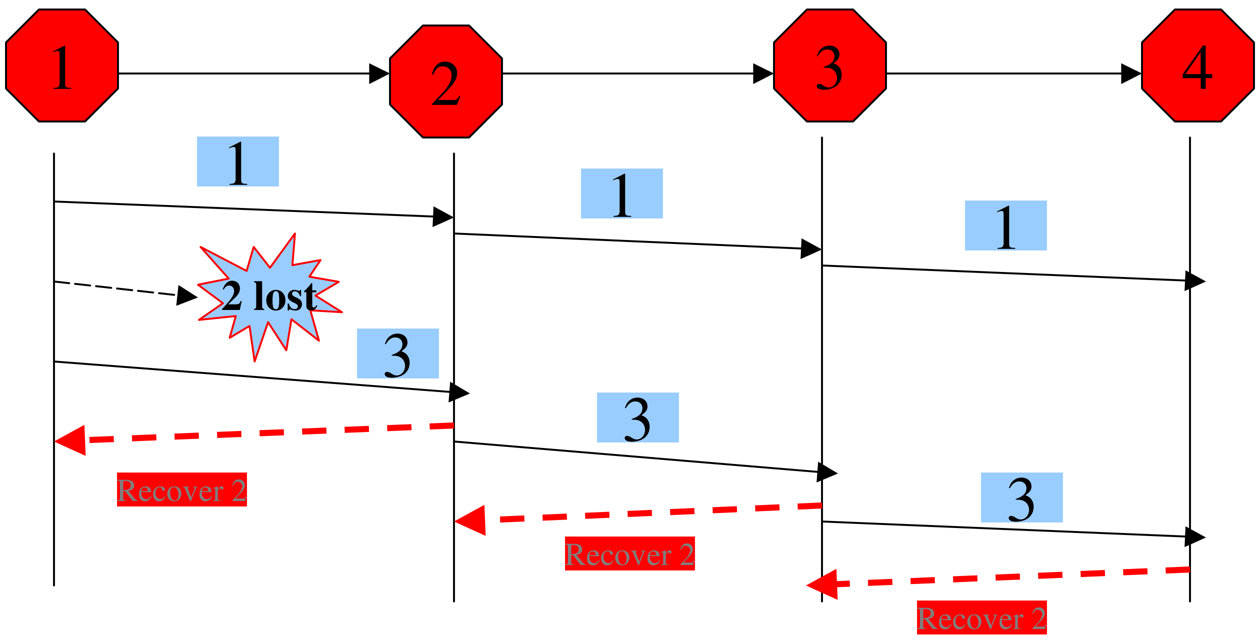
- 3 functions:
  - Pump: message relaying.
  - Error recovery: fetch.
  - Status reporting: report.
- Alternate between multi-hop forwarding when low error rates and store-and-forward when error rates are higher.

Multi-hop Packet Forwarding



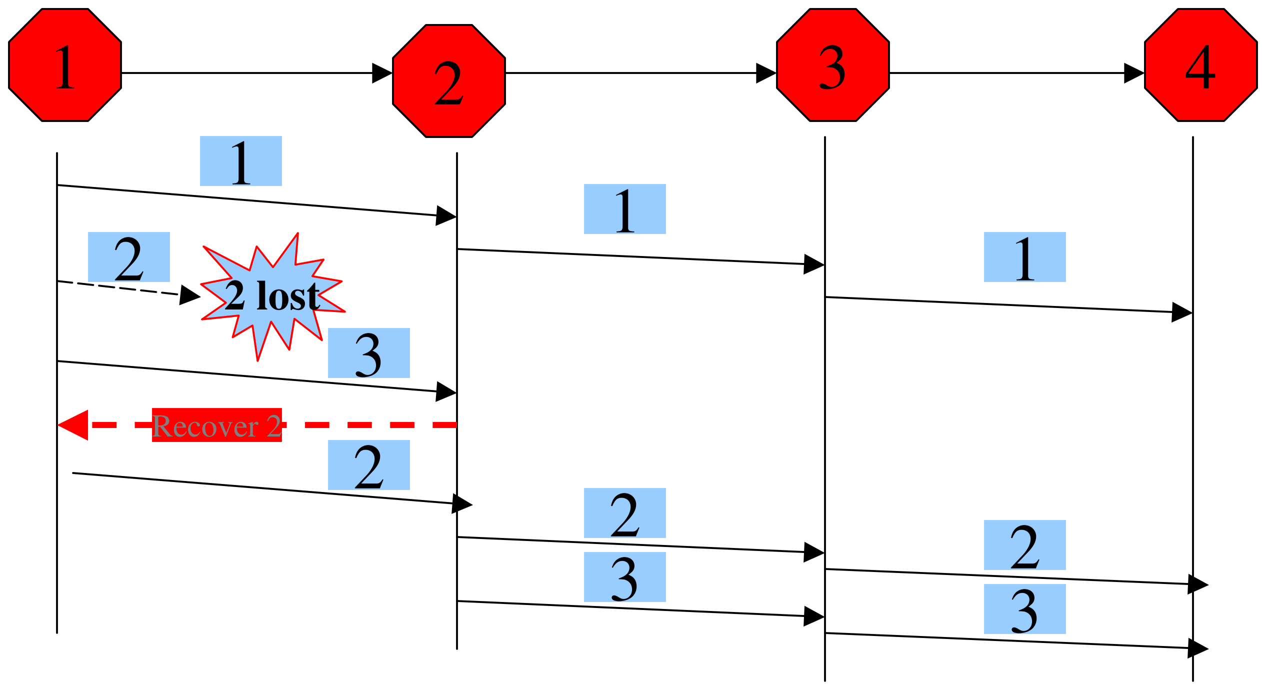
When no link Loss – multi-hop forwarding takes place

Recovering From Errors



Error recovery messages are wasted

PSFQ Recovers From Errors: “Store and Forward”

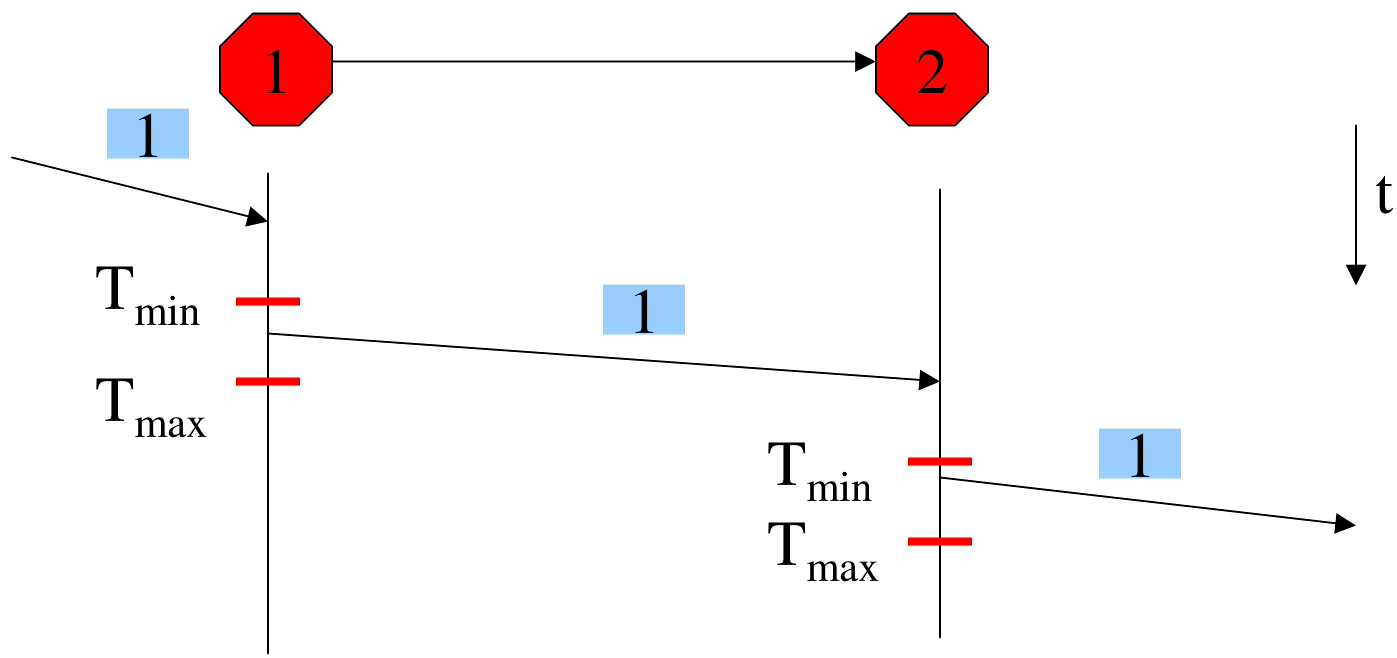


No waste of error recovery messages

Pump Operation

- Node broadcasts a packet to its neighbors every Tmin
  - Data cache used for duplicate suppression
- Receiver checks for gaps in sequence numbers
- If all is fine, it decrements TTL and schedules a transmission
  - Tmin < Ttransmit < Tmax
  - By delaying transmission, quick fetch operations are possible
  - Reduce redundant transmissions (don't transmit of 4 or more have forwarded the packet already)
  - Tmax can provide a loose delay bound for the last hop
    - ✓ D(n)=Tmax * n * N

PSFQ Pump Schedule



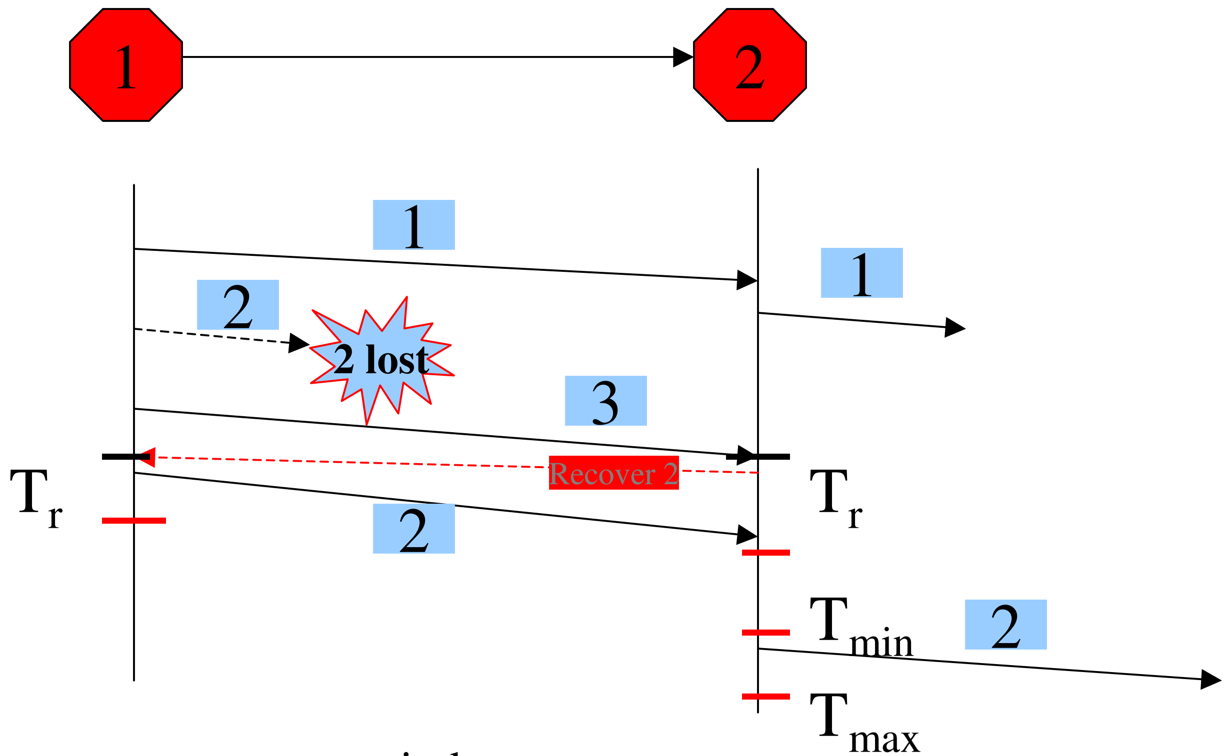
If not duplicate and in-order and TTL not 0 then
Cache and schedule for forwarding at time t (T_{min}<t<T_{max})

Fetch Operation

- Sequence number gap is detected
  - Node will send a NACK message upstream
    - ‘Window’ specifies range of sequence numbers missing
    - NACK receivers will randomize their transmissions to reduce redundancy
  - It will NOT forward any packets downstream
  - NACK scope is 1 hop
  - NACKs are generated every Tr if there are still gaps
    - Tr < Tmax
      - This is the pump/fetch ration
    - NACKs can be cancelled if neighbors have sent similar NACKs

Fetch Operation (cont'd)

When loss detected, then fetch mode.



Loss aggregation: try to recover a window of lost packets.

Proactive Fetch

- Last segments of a file can get lost
  - Loss detection impossible; no ‘next’ segment exists!
- Solution: timeouts (again)
  - Node enters ‘proactive fetch’ mode if last segment hasn’t been received and no packet has been delivered after Tpro
  - Timing must be right
    - ✓ Too early: wasted control messages
    - ✓ Too late: increased delivery latency for the entire file
  - Tpro = a * (Smax - Slast) * Tmax
    - ✓ A node will wait long enough until all upstream nodes have received all segments
  - If data cache isn’t infinite
    - ✓ Tpro = a * k * Tmax (Tpro is proportional to cache size)

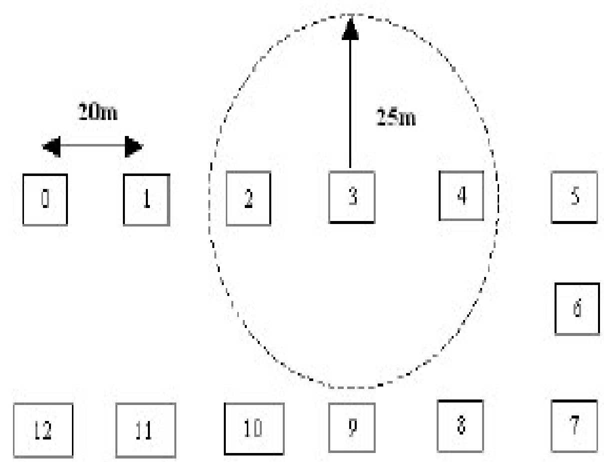
Report Operation

- Used as a feedback/monitoring mechanism
- Only the last hop will respond immediately (create a new packet)
  - Other nodes will piggyback their state info when they receive the report reply
  - If there is no space left in the message, a new one will be created
- Report aggregation.
- Carries status information: node id, seq. #.
- Triggered by user.
  - Inject data message with “report” bit set.

Performance Evaluation: Simulation

- Metrics
  - Average delivery ratio
  - Average latency
  - Average delivery overhead
- Selected application: network tasking
  - Radio: 2Mbps, 25 m range, simple CSMA/CA
  - Image file=2.5K, packet size=50 bytes (50 packets total)
  - Transmission rate: 1 packet/10 ms
  - Tmax = 100ms, Tmin = 50 ms, Tr = 20 ms
    - ✓ Fetch is 5 times faster than pump
- Comparison
  - SRM-I: SRM with an idealized omniscient multicast routing scheme

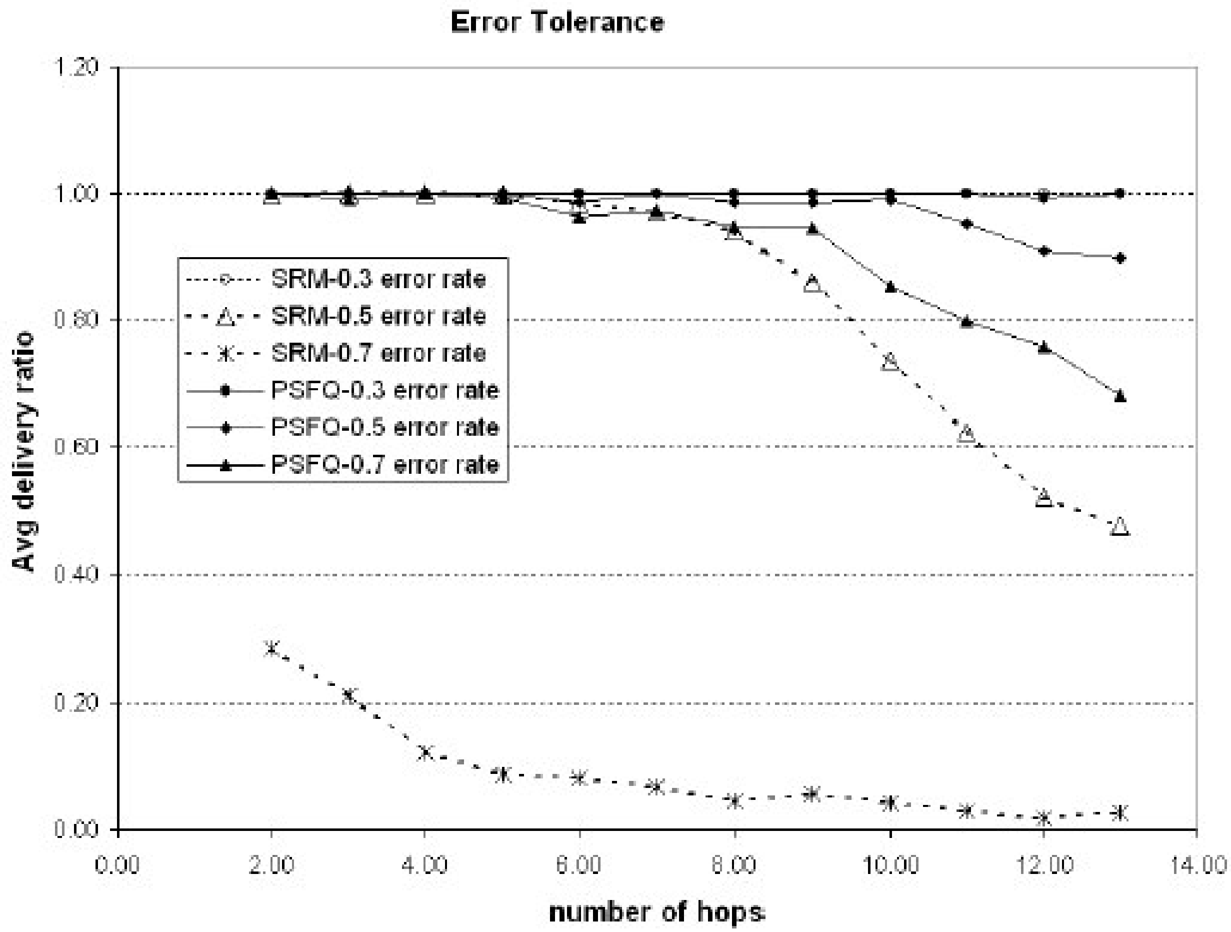
Simulation Setup



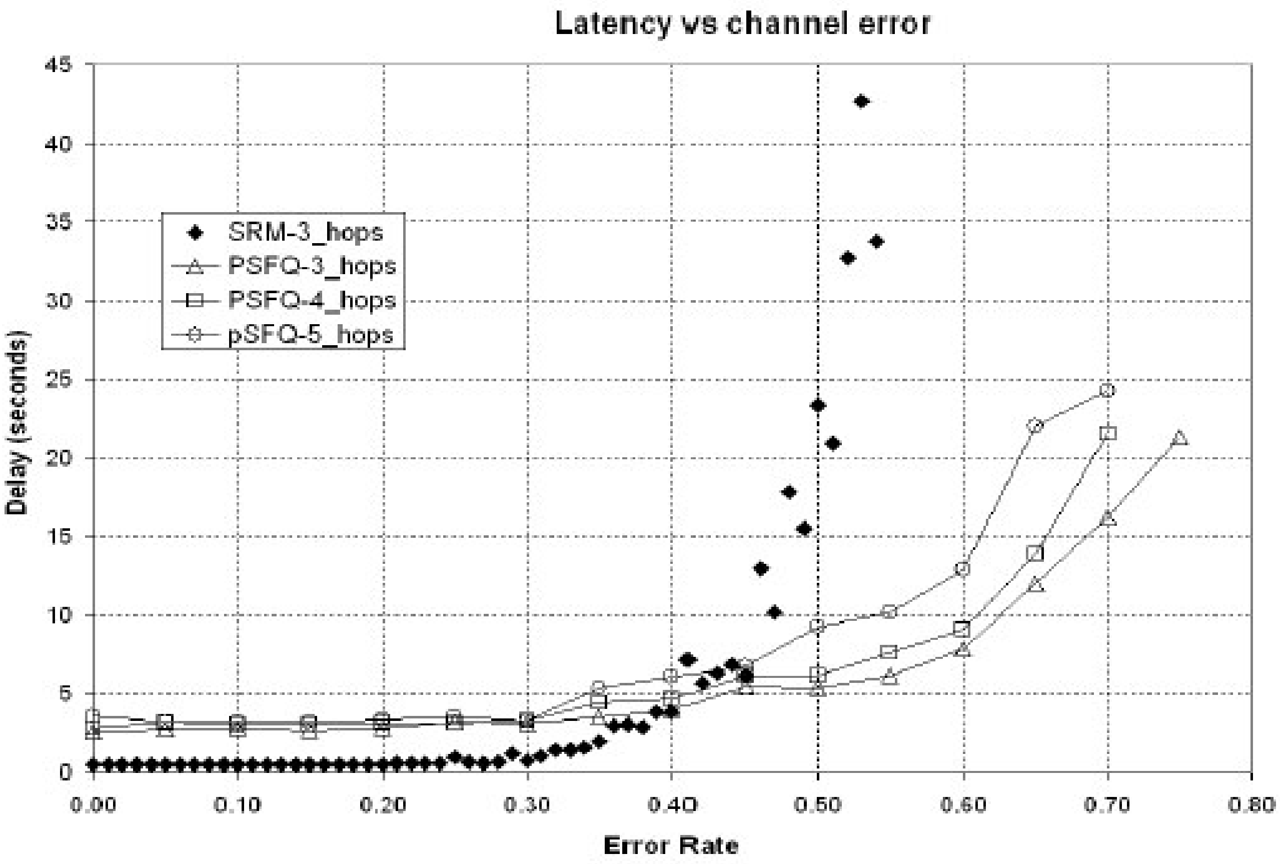
2 Mbps CSMA/CA Channel Access

T_{max} = 100ms T_{min} = 50ms T_r = 20ms

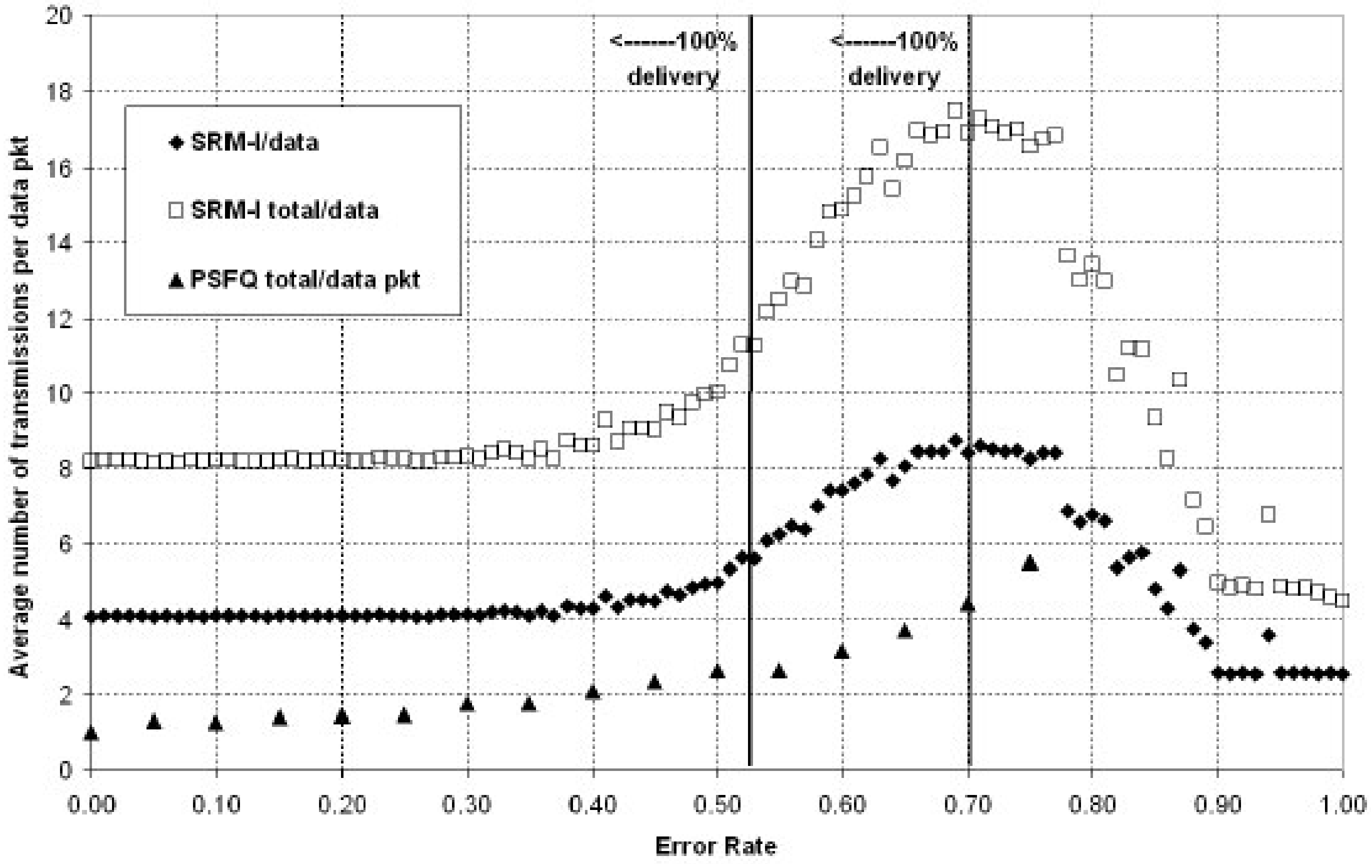
Error Tolerance



Average Latency

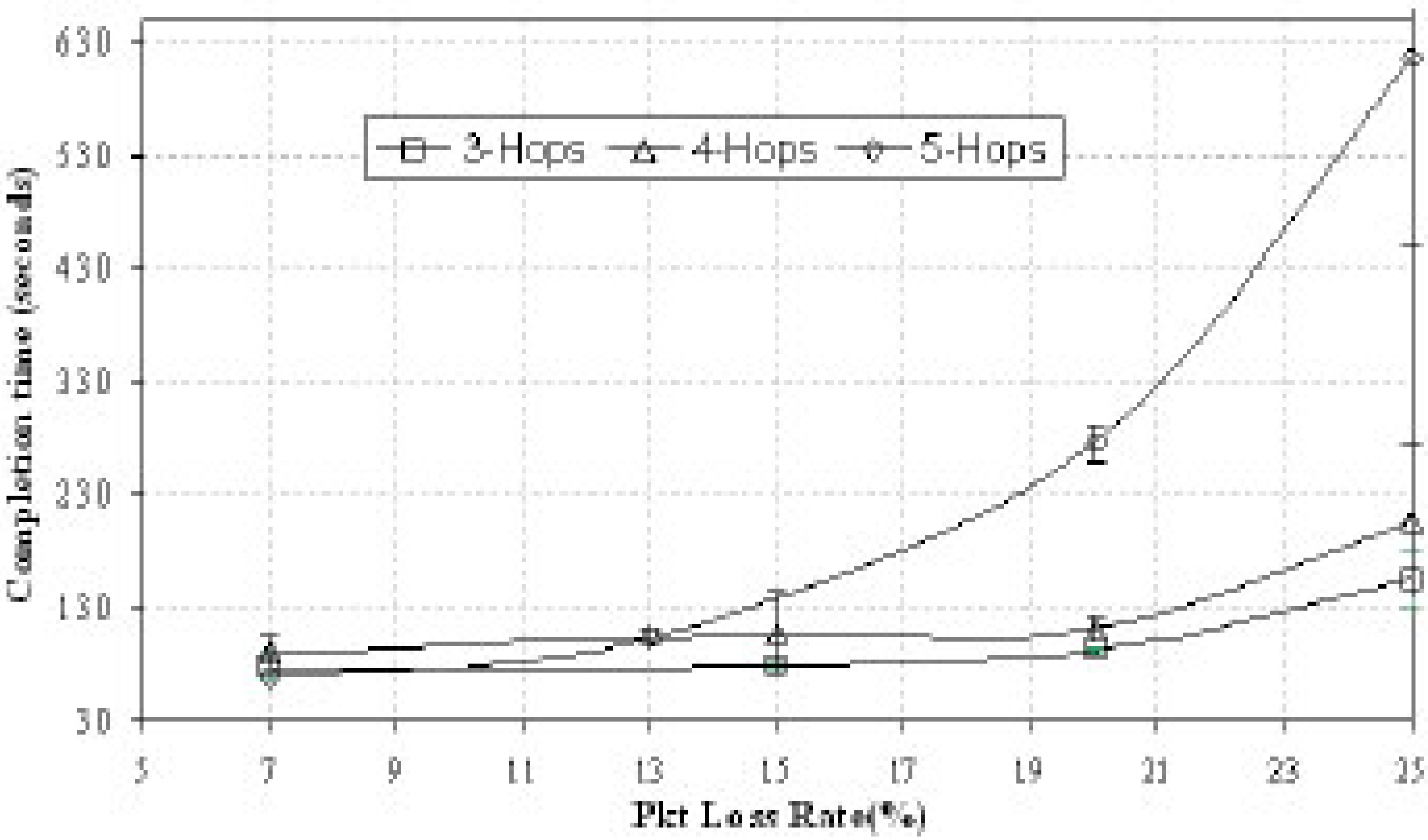


Overhead



<-------100% delivery
<-------100% delivery
SRM-I/data
SRM-I total/data
PSFQ total/data pkt
Average number of transmissions per data pkt
Error Rate
0
2
4
6
8
10
12
14
16
18
20
0.00
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

Experiment Results



- Much poorer than simulation: exponential increase in delay happens at 11% loss rate or higher
  - Was 35% for the 5-hop case in simulation

Conclusion - PSFQ

- Light weight and energy efficient
- Simple mechanism
- Scalable and robust
- Need to be tested for high bandwidth applications
- Cache size limitation

A Scalable Approach for Reliable Downstream Data Delivery in Wireless Sensor Networks

Seung-Jong Park,

Ramanuja Vedantham,

Raghupathy Sivakumar,

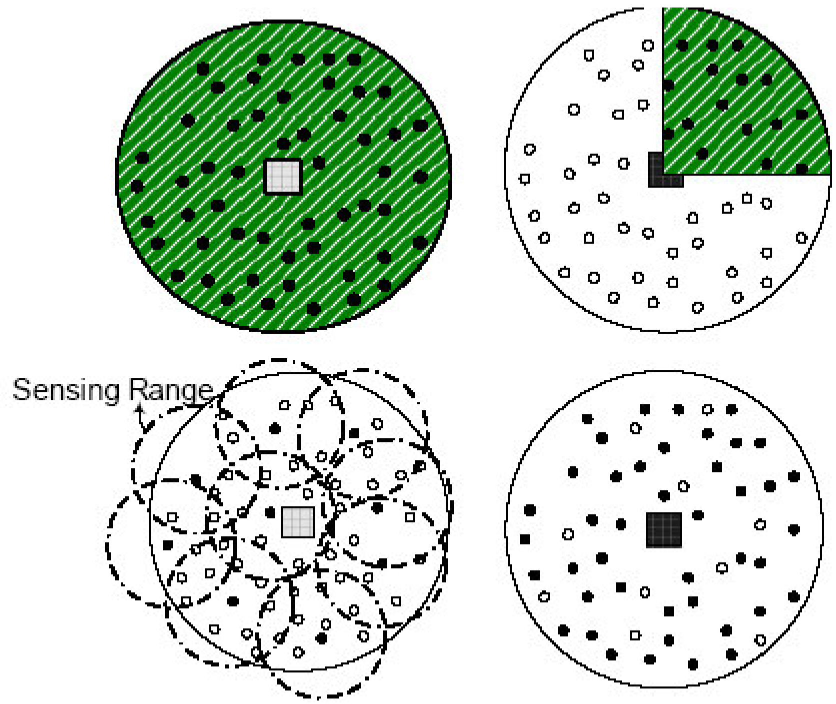
Ian F. Akyildiz

MobiHoc'04

Adopted from Seung-Jong Park's presentation at MobiHoc'04

Problem Definition

- A sink should deliver data to static sensors reliably
- Message considerations
  - Queries, Query-data, Control Code
- Scope of delivery considerations
  - Delivery to an entire area
  - Delivery to a sub-area
  - Delivery to the minimum # of nodes
  - Delivery to p% of nodes
- Environment considerations
  - Limited energy, low bandwidth, high node density, frequent node failures, no global node identification



Efficient loss recovery solution that addresses the above considerations

Design Preliminaries

- Packet forwarding
  - How to forward packets?
    - ✓ In-sequence [PSFQ] or out-of-sequence forwarding
    - ✓ Out-of-sequence forwarding for better spatial reuse
- Loss detection
  - How to request for lost packets?
    - ✓ ACK or NACK
    - ✓ NACK to avoid ACK implosion
- Loss recovery
  - Who and how to recover losses?
  - Local, designated scheme to decrease contention with packet forwarding

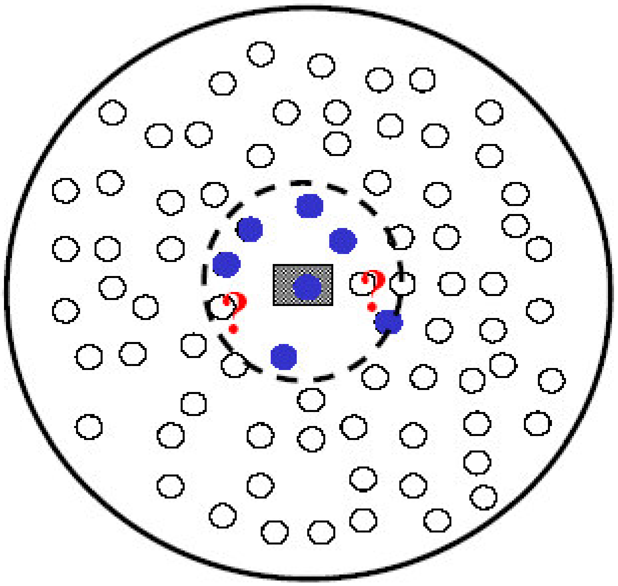
Design Challenges

- Single packet delivery
  - Reliably deliver single packet messages or small size messages
- Loss recovery
  - Determine an efficient recovery structure to recover losses
  - Determine when to request and recover lost packets
  - Prevent error propagation
- Reliable variants
  - Address the different reliability semantics

GARUDA: Accommodates the different considerations in a unified fashion while addressing the above challenges

Single Packet Delivery: The Problem

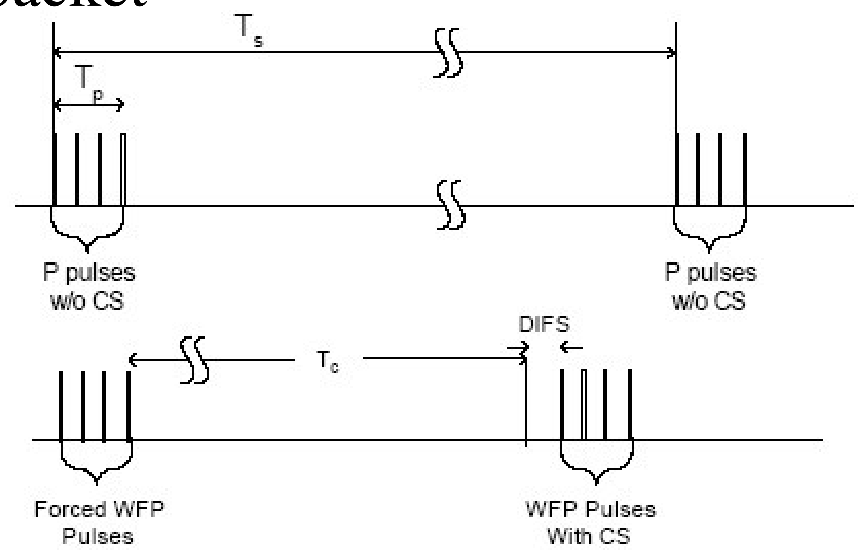
- For small messages or single packet messages
  - All the packets in a message can get lost
    - ✓ NACK cannot request for lost packets
  - ACK scheme results in ACK implosion
- Once the first packet reliability is supported, size of message is known
  - NACK can be used for requesting lost packets



To realize a scheme that supports first packet reliability

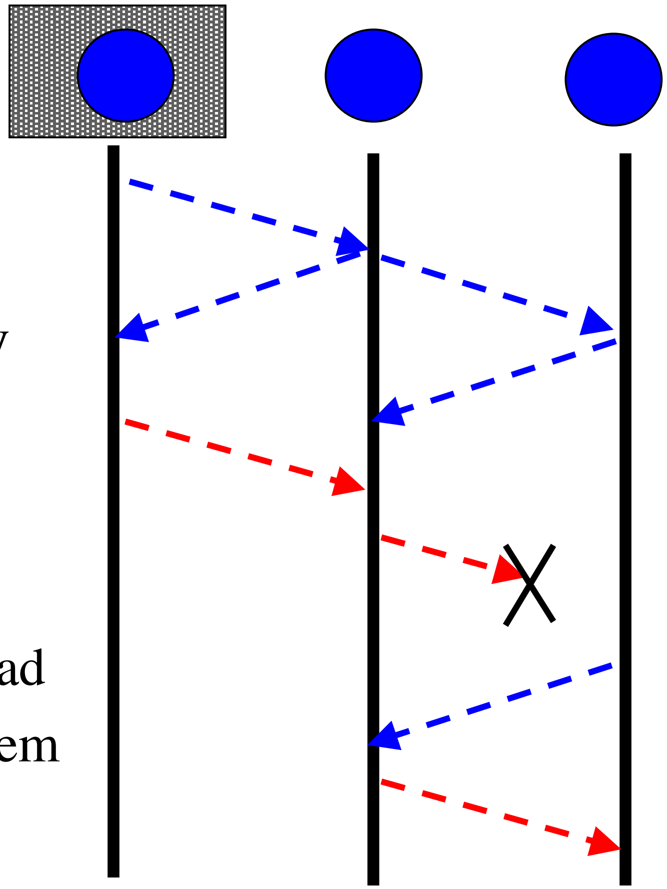
WFP Overview

- WFP (Wait-for-First-Packet) pulses
  - Used only for first packet reliability
  - Short duration pulses
  - Single radio
  - Advertisement of incoming packet
  - Negative ACK
  - Simple energy detection
- Different types of WFP
  - Forced pulses
  - Carrier sensing pulses
  - Piggybacked pulses



WFP Mechanism and Merits

- A sink sends WFP pulses periodically
  - Before it sends the first packet
  - For a deterministic period
- A sensor sends WFP pulses periodically
  - After it receives WFP pulses
  - Until it receives the first packet
- WFP merits
  - Prevents ACK implosion with small overhead
  - Addresses the single or all packet lost problem
  - Less energy consumption
  - Robust to wireless errors or contentions

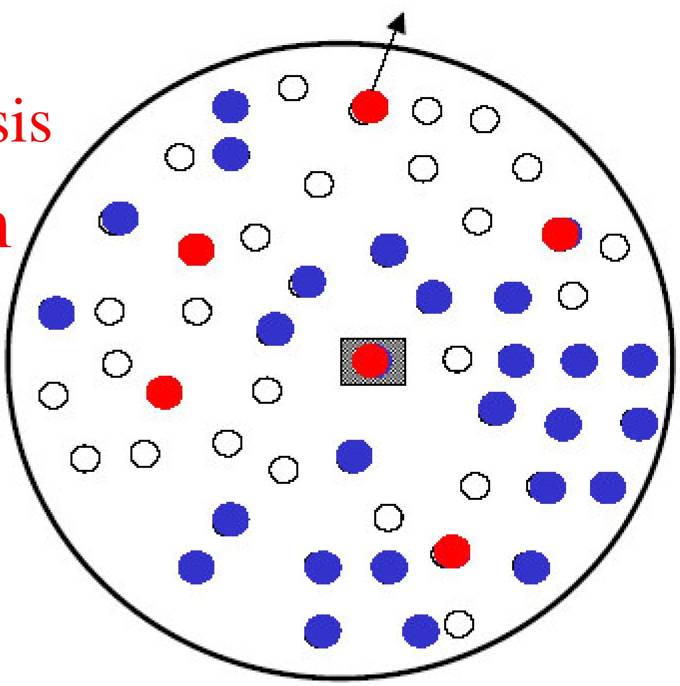


Loss Recovery: The Problem

- Designation of recovery servers
  - Construct the recovery server structure
    - ✓ Minimize the number of recovery servers
    - ✓ Low overhead and feasible designation
- Efficient loss recovery
  - Request for lost packets
    - ✓ Least possible contention with forwarding
    - ✓ Reduces the latency for recovery
- Error propagation
  - Out of sequence with NACK results in NACK implosion
    - ✓ Prevent propagation of NACKs

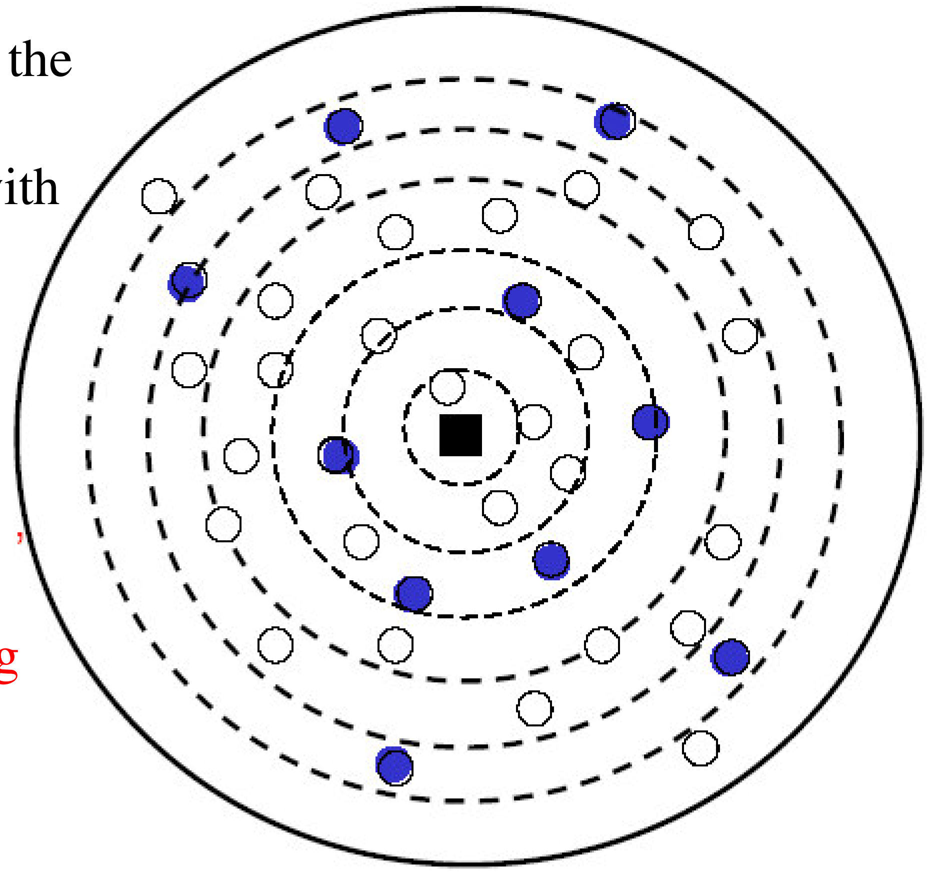
Recovery Server Designation

- Minimize the set of recovery servers
- Ideal solution: Minimum Set Cover (MSC)
  - Minimize the number of blue nodes selected to cover all white nodes
  - Infeasible because of per-packet basis
- GARUDA: Distributed Minimum Dominating Set
  - Approximation of MSC
  - Independent of loss pattern
  - Per message basis



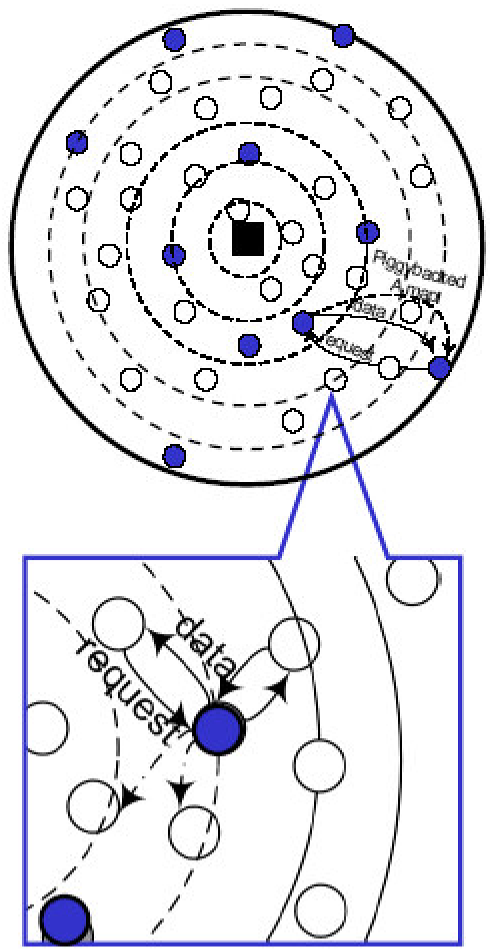
Core Structure

- Distributed MDS
  - Virtual bands constructed during the first packet flood
  - Core nodes chosen from nodes with band ID ***3i***
  - Adjacent nodes elected as core only if required.
- Core Merits
  - Approximation of the ideal solution , MSC
  - Decentralized construction during the 1st packet delivery
  - Fault tolerant
  - Low maintenance overhead



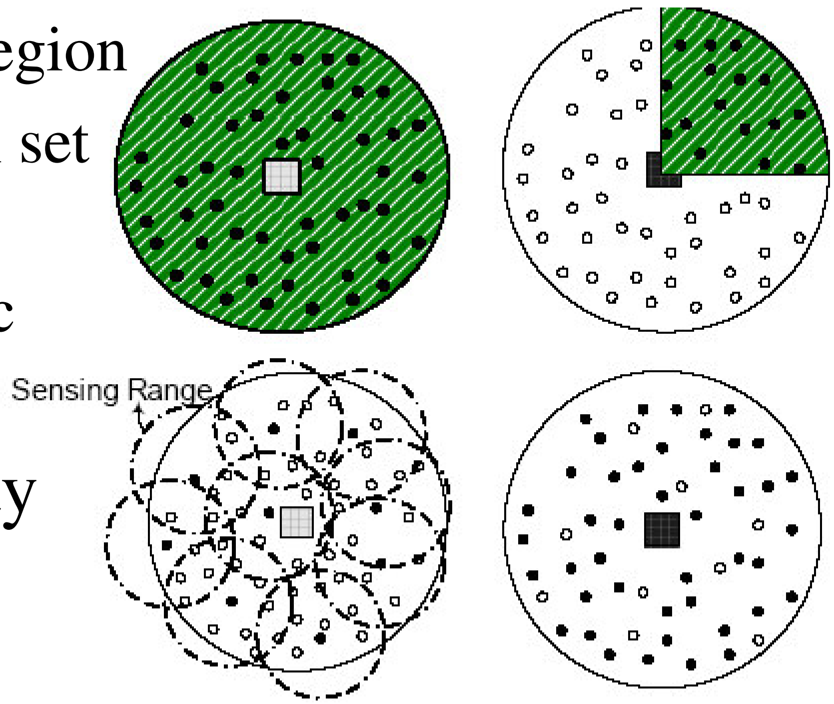
Two-Phase Loss Recovery

- Two-phase loss recovery
  - Phase 1
    - ✓ Loss detection and recovery between core nodes
    - ✓ At the end of phase 1, all core nodes receive all packets
  - Phase 2
    - ✓ Loss detection and recovery between non-core nodes and its core node
  - Availability-Map (A-map) is central in loss recovery
- Two-phase merits
  - Reduces the contention between loss requests and data forwarding
  - Reduces redundant retransmissions by utilizing wireless local broadcast



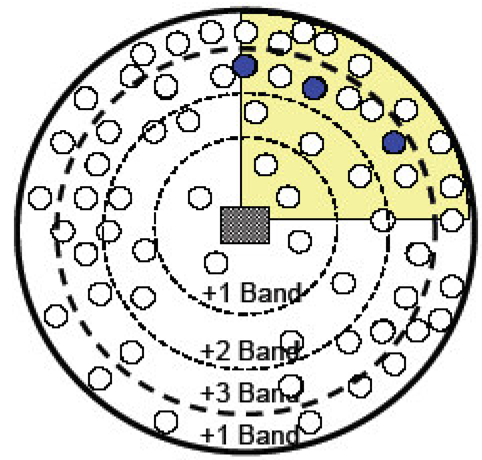
Variants: The Problem

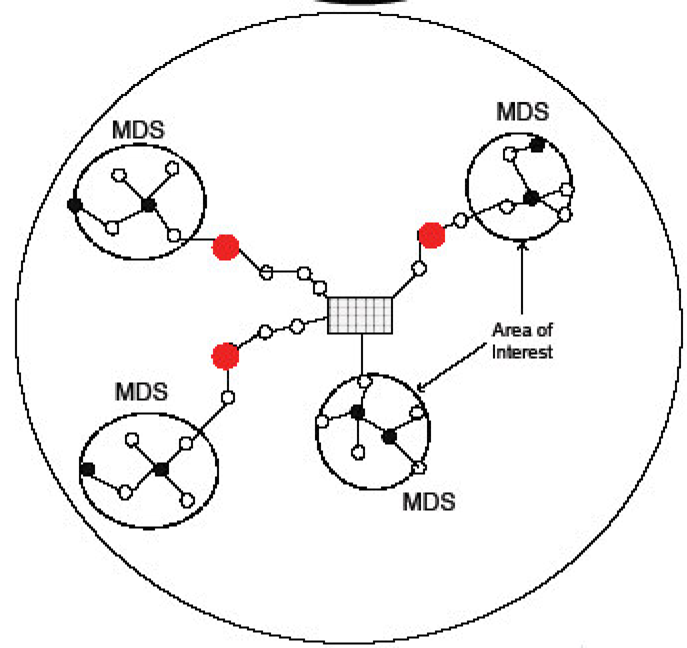
- How to address different types of reliability semantics
  - Reliable delivery within a sub-region
  - Reliable delivery to the minimal set of sensors
  - Reliable delivery to probabilistic subset
- Candidacy to address reliability variants
  - Easy extension to GARUAD



Candidacy

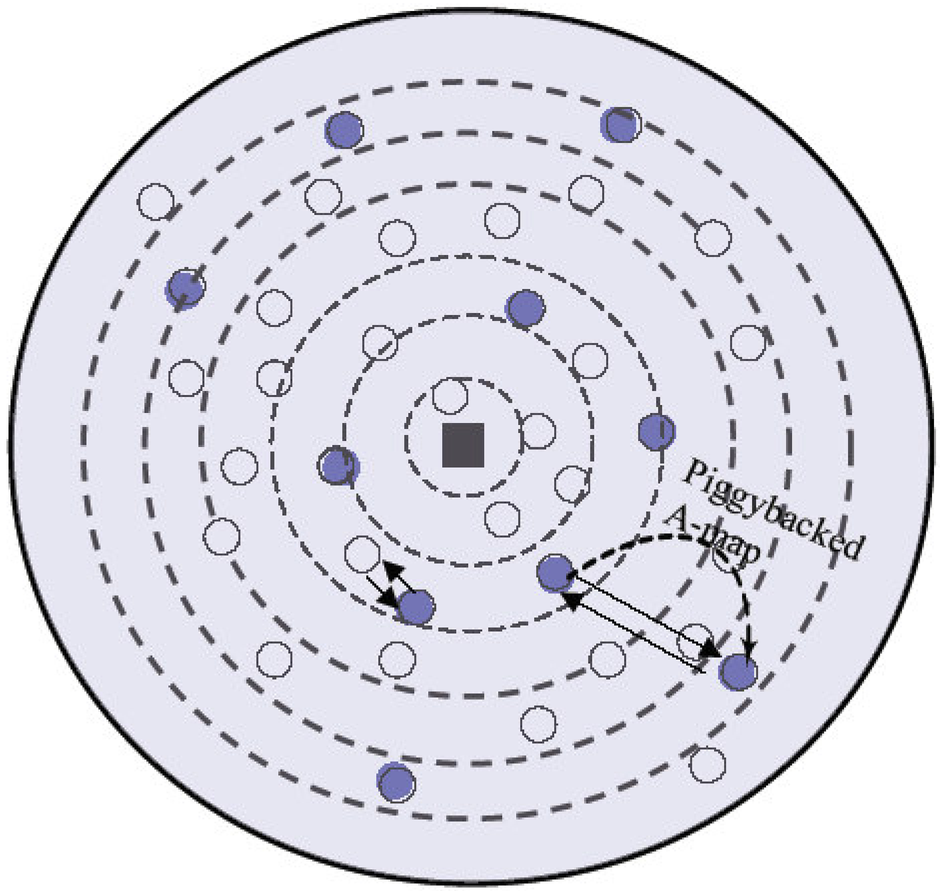
- Candidacy
  - Candidates chosen during first packet flood
- Core construction
  - Candidates participate in core construction
- Once core is established, use basic GARUDA
- If disjoint regions from sink
  - Forced candidacy
- Candidacy merits
  - Unified framework





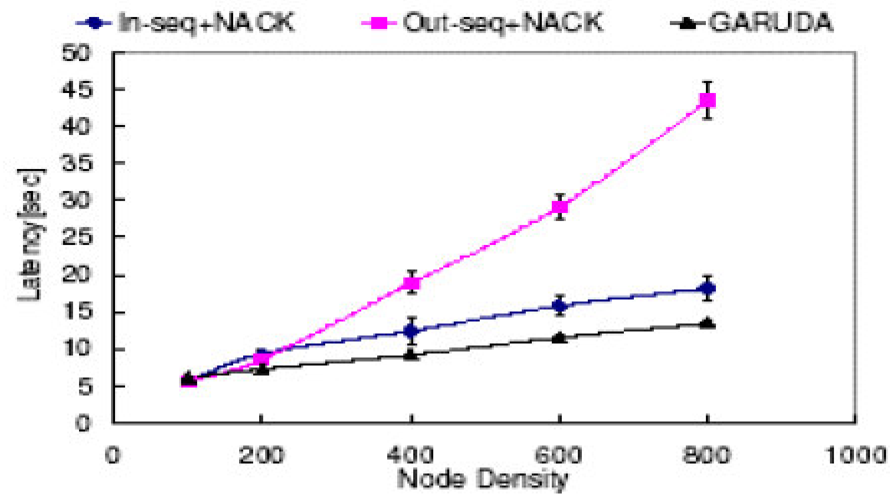
GARUDA Recap

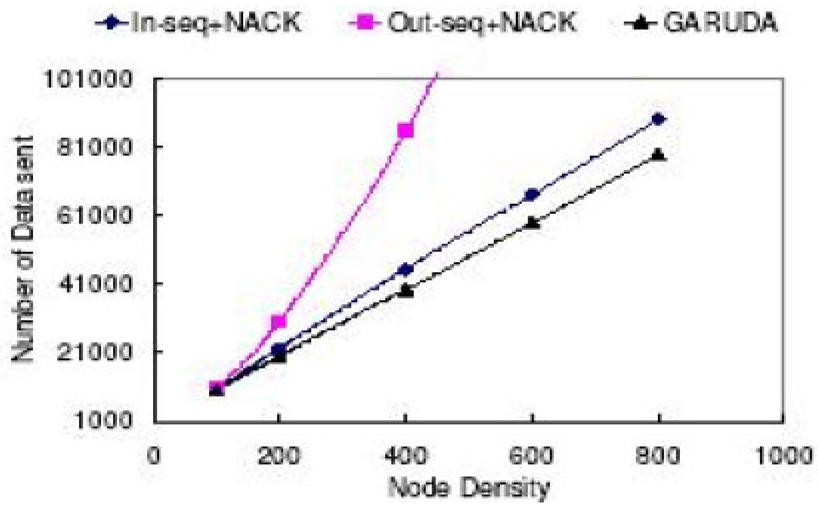
- Single packet delivery
- Candidacy
- Core construction
- A-map propagation
- Two-phase loss recovery

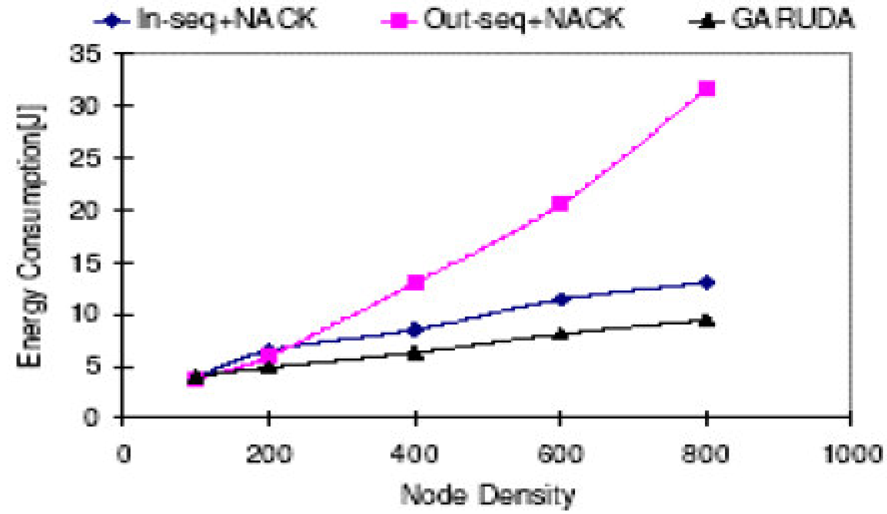


Performance Evaluation

- ❒ NS-2 simulation
- ❒ GARUDA performs better
  - ➢ Efficient core structure
  - ➢ Two-phase loss recovery
  - ➢ Availability map







Conclusion

- Presented a unified approach to handle message size consideration and scope of delivery
- Identified the ideal solution and the distributed approximation for ideal designation of recovery servers
- Demonstrated the effectiveness of GARUDA