

# Structured Communication in Single-Hop Sensor Networks

Amol Bakshi and Viktor K. Prasanna  
Department of EE-Systems  
University of Southern California  
Los Angeles, CA USA 90089  
{amol, prasanna}@usc.edu

1<sup>st</sup> European Workshop on Wireless Sensor Networks  
January 20, 2004

*Supported by US DARPA Power Aware Computing and Communication Program*



## Outline

---

- Motivation
- Related Work
- Contributions
  - Our System Model
  - $(N, p, k_1, k_2)$  routing in a single-channel WSN
- Recent Work
  - $(N, p, k_1, k_2)$  routing with **multiple** channels
  - Preliminary results
- Remarks



## Wireless Sensor Networks

---

- Sensor nodes have limited computation, communication and storage capacity
- WSNs are application-specific
  - The set of tasks and their general pattern of interactions are usually known at design time
  - Message exchanges are spatio-temporally correlated
- Communication in WSNs
  - Point-to-point communication is not the primitive
  - Protocol stack is application-specific
  - Most of current research involves manual optimization of protocols for specific communication patterns and performance metrics



## Structured Communication: What

---

- **Structured communication**
  - a routing problem where the pattern is known
  - examples: one-to-all, all-to-one, many-to-many, all-to-all, permutation, etc.
  - Other: spanning tree formation, depth-first traversal
- $(N, p, k_1, k_2)$  routing
  - $N$  packets to be transferred among  $p$  nodes
  - Each node transmits at most  $k_1$  packets
  - Each node receives at most  $k_2$  packets
- Permutation routing
  - $k_1 = N/p, k_2 = N/p$
  - Each node transmits **approximately** the same amount of information as it receives



## Structured Communication: Why

---

- Performance optimization
  - How much can *a priori* knowledge of communication patterns be exploited for efficient routing?
  - E.g: Load balancing to reduce/eliminate hot spots, reduce congestion and improve latency
  - Energy-balanced multi-hop routing (MWCN 2003)
- **Vision:** New method for application design
  - WSNs as distributed computing systems
  - Design efficient protocols for generic routing primitives and export a library of primitives to the end user and hide many low-level system details (MPI was a similar approach !!)
  - **Tradeoff:** Increased usability from the end users' perspective at the cost of reduced performance



## Single-hop ( $N, p, k_1, k_2$ ) routing

---

- Hierarchical, clustered topologies are energy-efficient
- Instances of ( $N, p, k_1, k_2$ ) routing can be useful within a cluster
  - Node readings are spatially and temporally correlated
  - Calibration, outlier detection, local data aggregation before transmitting information to other clusters or query node
  - Collaborative computation for distributed signal processing (e.g., target tracking, LOB estimation, etc.)



## Related Work

- *Energy-efficient permutation routing protocols in radio networks*, K. Nakano, S. Olariu, A. Zomaya, IEEE TPDS 2001
- *Fault-tolerant and energy-efficient permutation routing protocol for wireless networks*, A. Datta, IPDPS 2003.
- Reservation phase followed by data transfer phase
- All transfers are scheduled during reservation phase
- System model
  - Time is divided into equal size slots
  - Simultaneous transmissions in same slot cause collision
  - All nodes are time synchronized
- In-channel signaling: No distinction between latency and energy cost of control and data packets
- Both assume that nodes will NOT fail during the routing



## Related Work: Performance

**Table 1.** Simple Protocol in [4]

Nodes $p$	Packets $n$	%age Latency overhead	%age Energy overhead
50	1500	160	160
	2000	120	120
	2500	96	96
100	6000	163	163
	8000	122	122
	10000	98	98
200	20000	198	198
	30000	132	132
	40000	99	99

**Table 2.** General Recursive Protocol in [4]

Nodes $p$	Packets $n$	%age Latency overhead	%age Energy overhead
50	1500	293	300
	2000	295	300
	2500	96	100
100	6000	296	300
	8000	297	300
	10000	98	100
200	20000	298	300
	30000	298	300
	40000	99	100

- Latency overhead:  $(T - n) / n$
- Energy overhead:  $(E - 2n) / 2n$
- High overhead due to small size network and in-channel signaling; can be reduced on 2-channel architecture
- **Did not consider energy balance of the routing**



## Our Model

---

- All nodes are homogeneous, similar clock frequency, single-hop connectivity
- All nodes are time synchronized
- Two radios per node: data and control
  - Two-channel architectures are being designed: e.g. PASTA project (USC, ISI, UCB, UCLA, MIT)
- Comparison of data/control costs
  - RFM TR1000 vs. PicoRadio: energy to Xmt/Rcv one data packet is around **two orders of magnitude** more than control packet
  - S-MAC RTS/CTS (32B), USC/ISI Mica Stack (max. 250B): time to Xmt/Rcv one data packet is **one order of magnitude** more than control packet



## Basic Technique

---

- Medium access
  - Decentralized TDMA-like coordination over control channel
  - Data transfers triggered by outcome of transactions on control channel
- *A priori* scheduling of control channel
  - First  $N/p$  control slots owned by node 0, next  $N/p$  slots by node 1, and so on
  - Only the owner of a control slot transmits an RTS
  - Successful RTS/CTS (no node failure) leads to immediate data transmission over data channel
- In our routing primitive
  - Each node knows destination IDs of its packets
  - No node knows the source IDs of packets for itself
  - Each node transmits  $N/p$ , receives  $N/p$



## Contributions

---

- Simulate TDMA-like coordination using RTS/CTS and contention-based MAC protocol
- Tolerate transient and permanent node failures
- Use RTS/CTS to dynamically compress the schedule to save latency and energy in case of faults
- Robust: all packets transferred directly from src to dest
  
- *A priori* knowledge of communication pattern is required and exploited!
- Cost overheads will depend entirely on radio parameters and packet sizes (state-of-the-art numbers are encouraging)
  
- Objective: To evaluate if simple protocol is good enough for 'reasonably' sized sensor networks



## Protocol Phase I: Pre-processing

---

- Similar to Nakano et al (IEEE TPDS 2001)
- Executed entirely over the control channel
- $p$  rounds: in slot  $i$  of round  $j$ , node  $j$  transmits the number of packets it has for node  $i$
- At the end of  $p^2$  slots, all nodes know their value of  $k_2$  – in case values are different for different nodes
- This is required for nodes to turn themselves off once they have transmitted  $k_1$  and received  $k_2$



## Protocol Phase II: Data Transfer

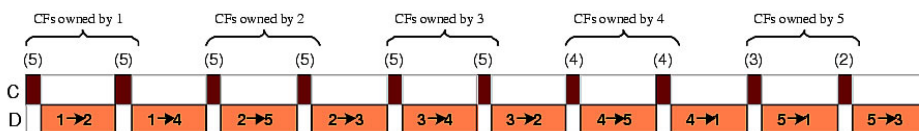
```

If I own the current CF[i]:
  Remove next packet from transmit queue
  Send RTS to destination
  Wait for CTS from destination
  Transmit packet in DF[i]
Else:
  If I have received N/p packets:
    Remain ShutDown
  Else:
    Listen to RTS sent by owner of CF[i]
    If I am the destination:
      Reply with a CTS
      Receive packet in DF[i]
    Else:
      Sleep in DF[i]

```



## Protocol Illustration



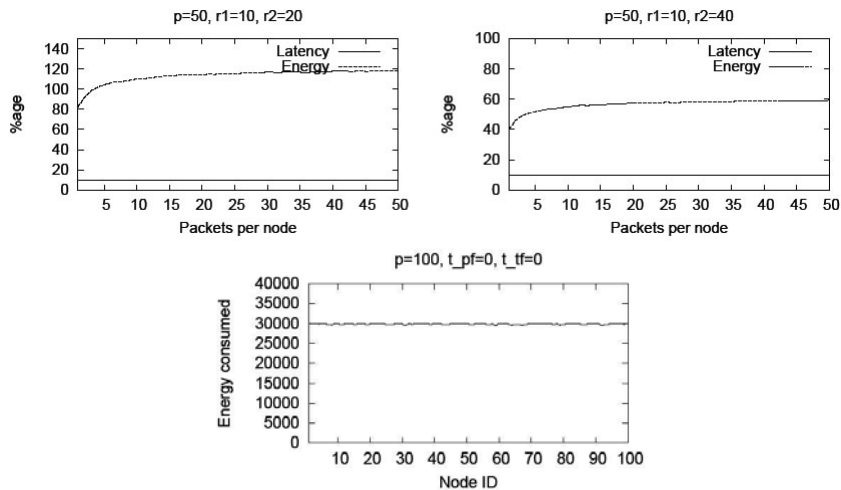
$p = 5, N = 10$ . Packet distribution: 1: [2,4]. 2: [5,3]. 3: [4,2]. 4: [5,1]. 5: [1,3]

C denotes the control channel (upper); D denotes the data channel (lower)

Numbers in brackets above the CFs denote number of nodes monitoring the channel in that CF



## Latency/Energy Overhead: No Faults



- Overhead is the **cost of robustness** (Lower bound is NOT 0%)
- Latency overhead is constant

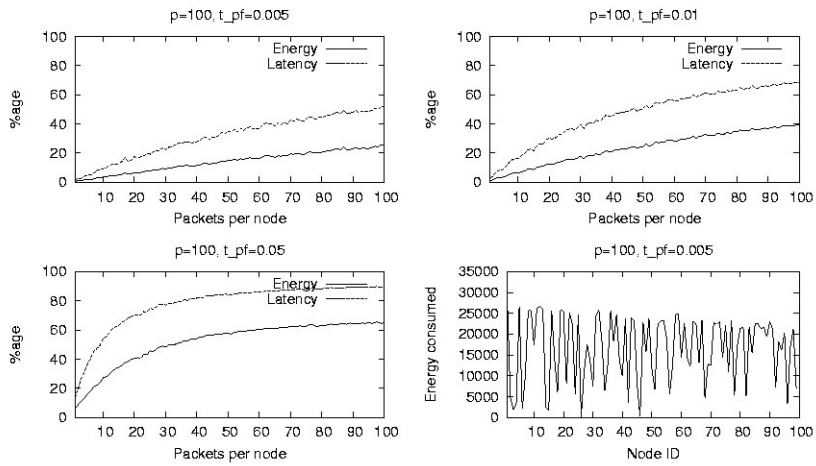


## Handling Communication Failures

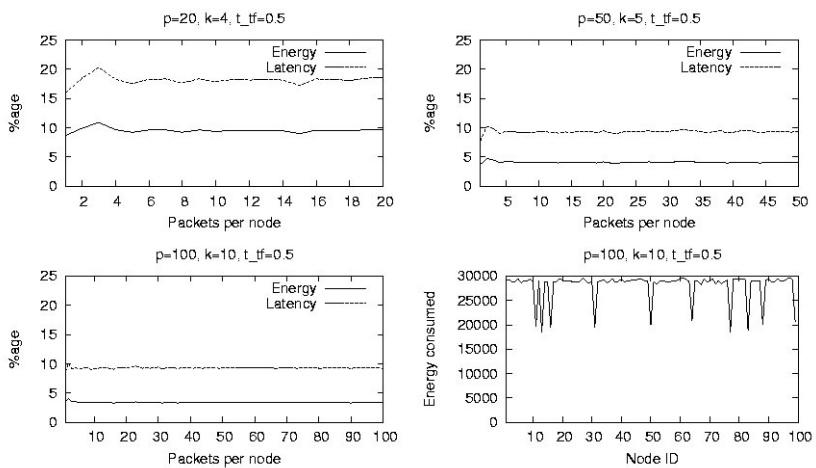
- Two types of communication failures
  - **Permanent**: Once failed, node will remain failed
  - **Transient**: Node can become available again
- All nodes who have not received packets are listening on the control channels
- If RTS or CTS is not heard
  - Permanent failure
    - data transfer is canceled, schedule is compressed, and next RTS/CTS follows immediately
    - Node is removed from consideration for all future transactions
  - Transient failure
    - No data transferred in slot
    - Schedule is not compressed since node can come up!



## Permanent fault model: Savings

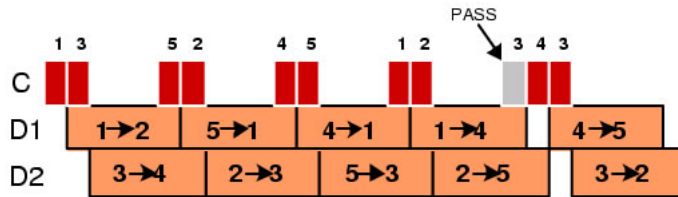


## Transient fault model: Savings





## Extension to Multiple Channels

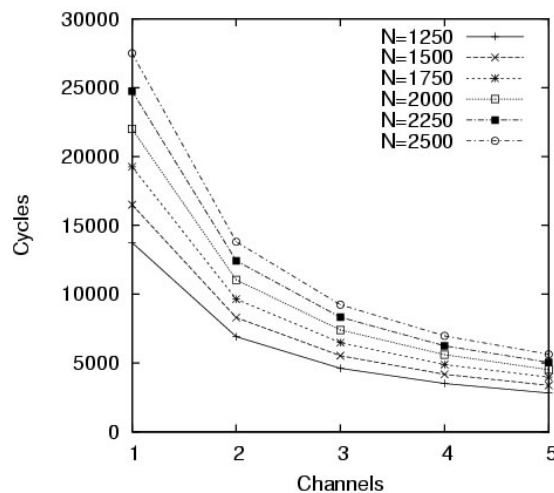


$p = 5, f_d = 2, N = 10$ . Packet distribution: 1: [2,4]. 2: [5,3]. 3: [4,2]. 4: [5,1]. 5: [1,3]  
C denotes the control channel; D1 and D2 denote the data channels

- One control channel, multiple data channels
- Same basic idea as single-channel protocol
- Find matching of communication graph dynamically!



## Preliminary Results: 50 nodes





## Remarks

---

- Objective was to evaluate if a simple contention-based protocol is 'good enough'
- Protocol performance is good for single-hop networks with few tens of nodes (real-world?)
- Using a combination of (a) RTS/CTS as the basic coordination mechanism, and (b) a priori knowledge of communication pattern, allowed easy extensions for fault tolerance