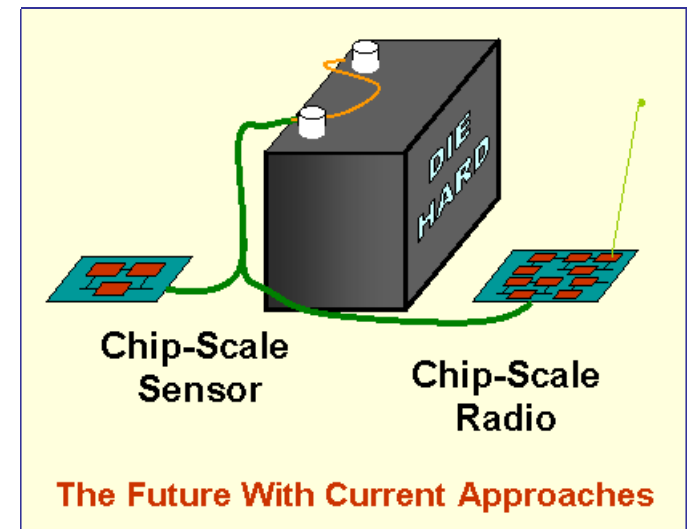
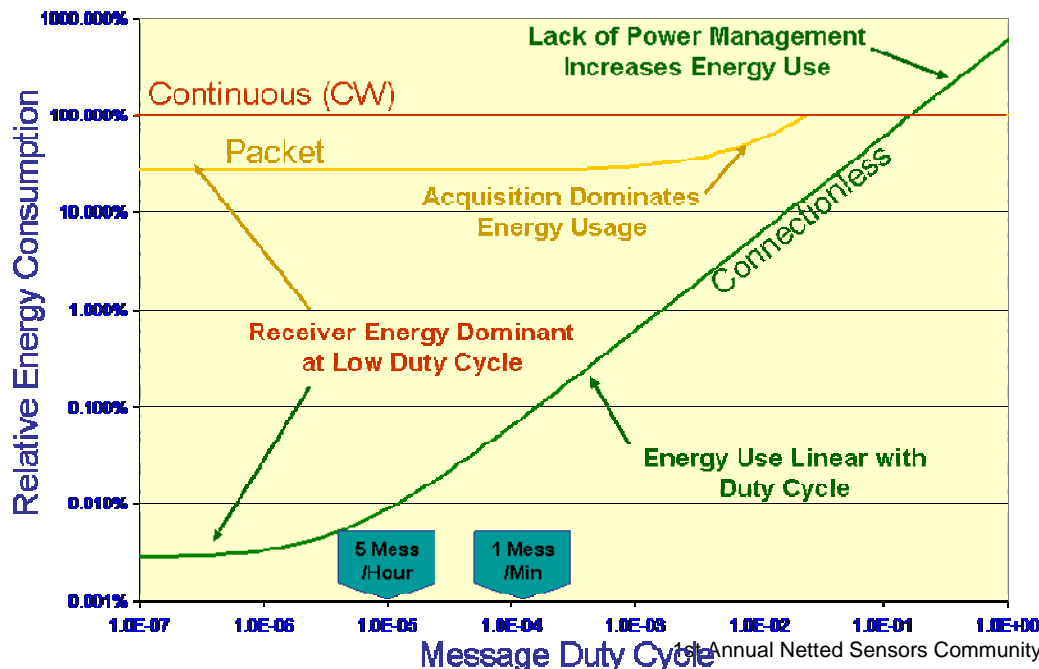


Communications and Networking Session: Cross Layer System Design for Sensor Networking

Dr. Jason Redi
BBN Technologies
Cambridge, MA
redi@bbn.com

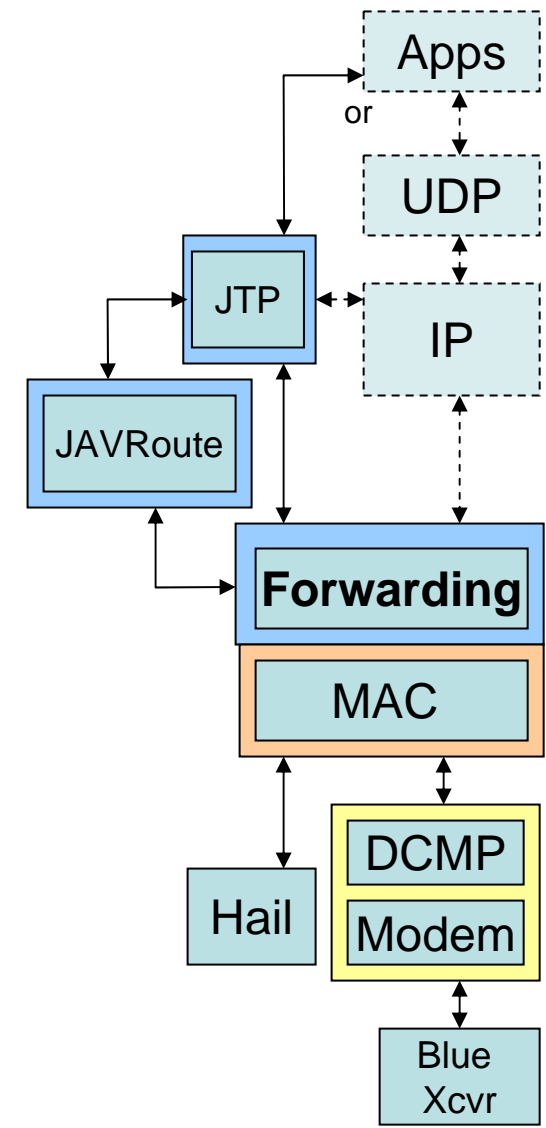
DARPA Connectionless Networks Program

- Size/Weight/Power of Sensing and Digital Components Continues to Drop
- Energy Required to Send a Bit Remains Constant
 - Driven by Shannon's Law and Physics
- Energy Required by RF Link and Network Topology Limits Lifespan, Miniaturization, Covertness, ...
- Systems designed for high offered load are energy inefficient at low loads
 - But we need high offered load *too*



JAVeLEN

- Joint Architecture Vision for Low Energy Networking (JAVeLEN)
- Two Example Methods for Cross-Layer Sensor Networks
 - End-to-End Transport protocol with mid-hop operation
 - Dynamic Time Synchronization



Observations about Multi-hop

- **Wireless BER is so bad that we use P2P-ACKs for unicast packets.**
 - We can *overload* P2P-ACKs to handle some error cases (e.g. queues full).
- **A to B and B to A paths are usually the same**
 - Therefore E2E Packets and E2E-ACKs will flow over the same nodes.
 - If path does not change, any retransmissions will flow over the same nodes and the original transmission.
 - Useful to cache packets in mid-path to response to E2E-NAKs early in the path.
- **Things the Network “knows”:**
 - Reachability
 - Can help with establishment and disconnection indication.
 - Path length
 - Can help with initial determination of bandwidth-delay product.
 - Hop-by-Hop costs
 - JAVeLEN estimates energy and link utilization and reports that as hop-by-hop costs.
 - Can help with estimate of non-congestion wireless packet losses.
 - Network Topology
 - Can help with identifying backup or alternative paths.
 - Any of this information might still be temporarily wrong or changing.

Reconsidering the need for an End-to-End protocol

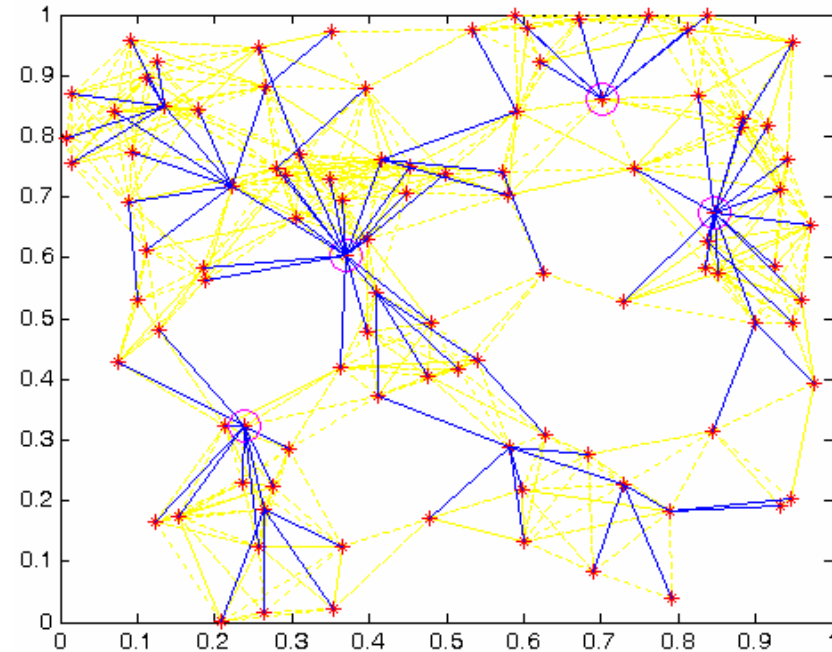
- **“A funny thing happened on the way to the destination...”**
- **Exceeded retransmissions in mid-path**
 - No CTS (for RCDA) or no ACK (for DA only)
 - Might mean that node is no longer there
 - Got CTS, but no ACK
 - Node is there, but congestion likely
 - Can wait and try later (but not too long) OR
 - Try an alternative path
 - Nodes can measure local RF energy (if non-spread), correlation (if spread), and promiscuous packets to estimate if area is typically “busy”.
- **Arrived at node in mid-path OK, but:**
 - No next-hop
 - Path to destination lost during packet’s travels
 - Can send along alternative route (source route) OR,
 - Can send explicit NAK back to source (assuming path to *source* still exists)
 - No space in queues
 - Solution – don’t allow packets if no space available. Use a NAK packet and push problem back to previous hop.
 - Either wait, send along alternative path, or send NAK back to source.

Solutions to E2E Packet Losses

- **Arrived at node in mid-path OK, but: (con't)**
 - Node destroyed or rebooted
 - Everything seems OK, but packets just never gets past one particular hop.
 - Only solvable with E2E acknowledgements
 - Internal error in node (surprisingly common – [StonePartridge01])
 - Hardware or software pointer corrupted, Bus timing errors, etc.
 - Only solvable with E2E acknowledgements
- **Summary**
 - Need both E2E-ACKs and P2P-ACKs
 - P2P-ACKs deal with lousy wireless BER, E2E-ACKs deals with errors P2P can't catch
 - Send E2E ACK
 - Have Non-ACK'd data and no new data is received for a period of time or
 - Received X packets without sending E2E-ACK.
 - Sequence numbers indicate missing data for more than 2x expected delay.
 - End of stream indication received.
 - Overload P2P-ACKs
 - Provide Data NAK to indicate no route, queue full, etc.
 - Allows previous hop to make alternative decision.
 - Overload E2E-ACKs
 - Informs source to stop injecting data.
 - Informs nodes along the path to drop packets (and caches) to the dest.

Dynamic Time Synchronization

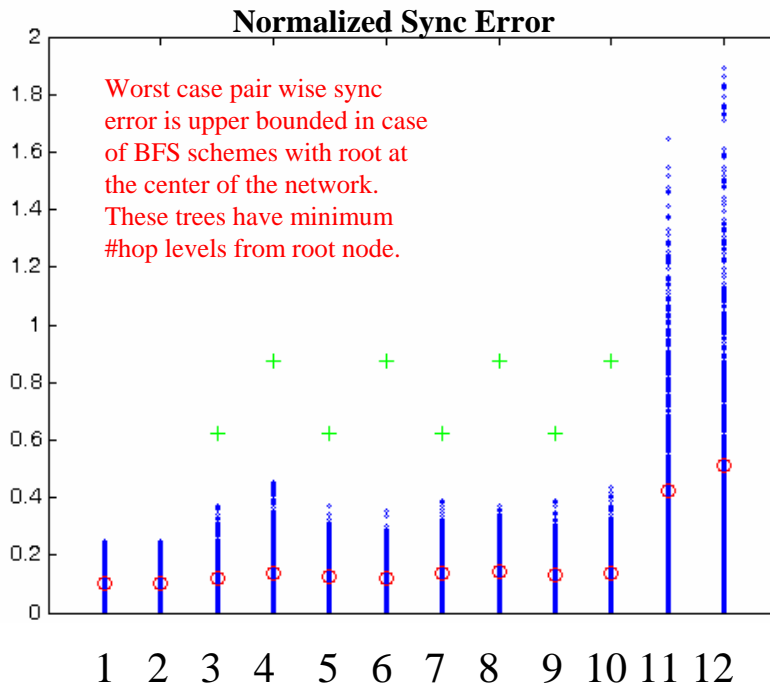
- **Background:** Low offered load environments need sync for shutting down receivers according to slots.
- **Problem:** Real clocks have time drift
 - 1ppm clock in 100kbps radio drifts 0.1 bit time per second, two nodes drift 120kbps in a week.
- **Solution:**
 - Dynamic time synchronization
 - Election of leader/root
 - Computation of energy efficient “sync-tree”
 - Slot-alignment messages along sync-tree
 - Maintenance of tree and slot re-synchronization



Circled nodes are “local roots”
One among them will be elected the leader

Sync-trees

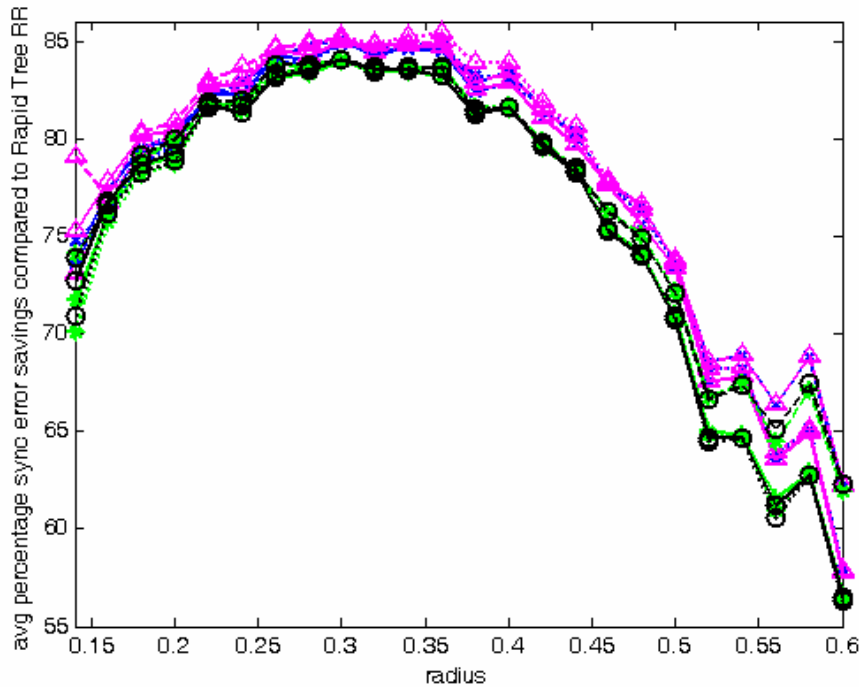
- Sync-tree : a spanning tree of the network topology graph
- Several optimality considerations
 - Minimize worst case pair wise sync error between any two neighboring nodes in the network topology (influences guard time provisioning)
 - Minimize number of transmitting nodes in the tree (Should minimize the #slot-align message transmissions during later phases of the protocol.)
- It is hard to construct a tree that satisfies all the above properties → Use heuristics



- 1: Min-distance Dijkstra (center root)
- 2: Min-distance Dijkstra (random root)
- 3: Min-hop Dijkstra (center root)
- 4: Min-hop Dijkstra (random root)
- 5: BFS (center root)
- 6: BFS (random root)
- 7: BFS node-sorted (center root)
- 8: BFS node-sorted (random root)
- 9: BFS edge-sorted (center root)
- 10: BFS edge-sorted (random root)
- 11: Rapid (center root)
- 12: Rapid (random root)

Performance Evaluation

% savings in worst case sync-error of BST trees over rapid trees



% savings in number of transmissions for slot-align messages of BST trees over rapid trees

