Cross-Layer Designs for Energy-Saving Sensor and Ad hoc Networks

Matthew J. Miller

Preliminary Exam

July 22, 2005

A Tale of Two Network Stacks

It was the best of designs,…

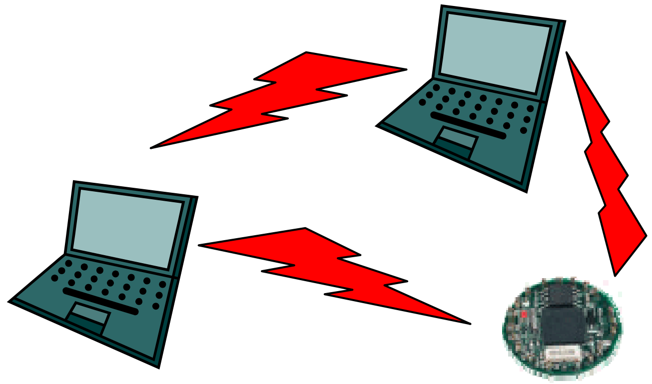
Application

Transport

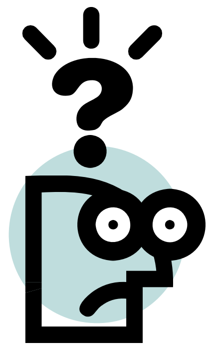
Network

Data Link

Physical



…it was the worst of designs.



Application

Transport

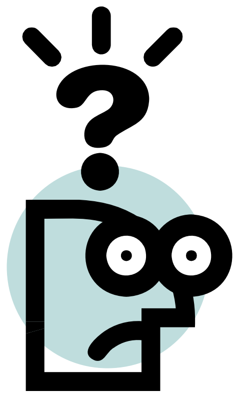
Network

Data Link

Physical

Why not strict layering?

- Why shouldn't wireless sensor and ad hoc networks use the principle that has worked so well for wireline networks?
  - Network usage
  - Network performance



Rethinking the Design: Network Usage

Wireline/Internet

- Connection Oriented
  - The network gets data from point A to point B
- General purpose
  - Same architecture for email, streaming video, and large file downloads

Ad hoc/Sensor

- Task Oriented
  - The network performs specified task
- Specific usage
  - Habitat monitoring and intruder detection may have very different requirements at multiple layers

Rethinking the Design: A Lesson From Business?

From Christensen and Raynor's *The Innovator's Solution*:

- "When products are not yet good enough, companies should set up a proprietary, in-house architecture to capture the most profits."
  - Cross-layer interactions to improve performance
- "When products become more than good enough, commoditization sets in and activities should be outsourced."
  - Modularization to focus on core component design

Rethinking the Design: Network Performance

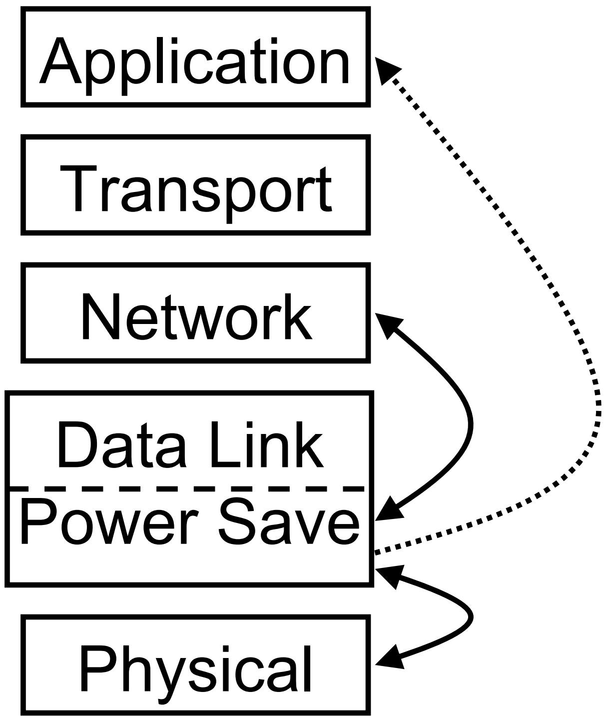
Wireline/Internet

- Relative performance is "good enough"
  - Modularization and cleaner interfaces
- Lower layer behavior well-defined
  - TCP timeouts
  - Link loss
  - Re-establishing route

Ad hoc/Sensor

- Performance rarely "good enough"
  - Needs cross-layer interactions to improve performance
- Lower layer behavior unknown
  - Setting timeouts?
  - Differences in "links"?
  - How expensive is route discovery?

Our Contribution to Cross-Layer Design and Interactions



Cross-layer design and interactions for energy efficient protocols

- Link layer/physical layer designs
- Link layer/network layer designs
- Effects and tradeoffs on applications from energy saving protocols

Talk Outline

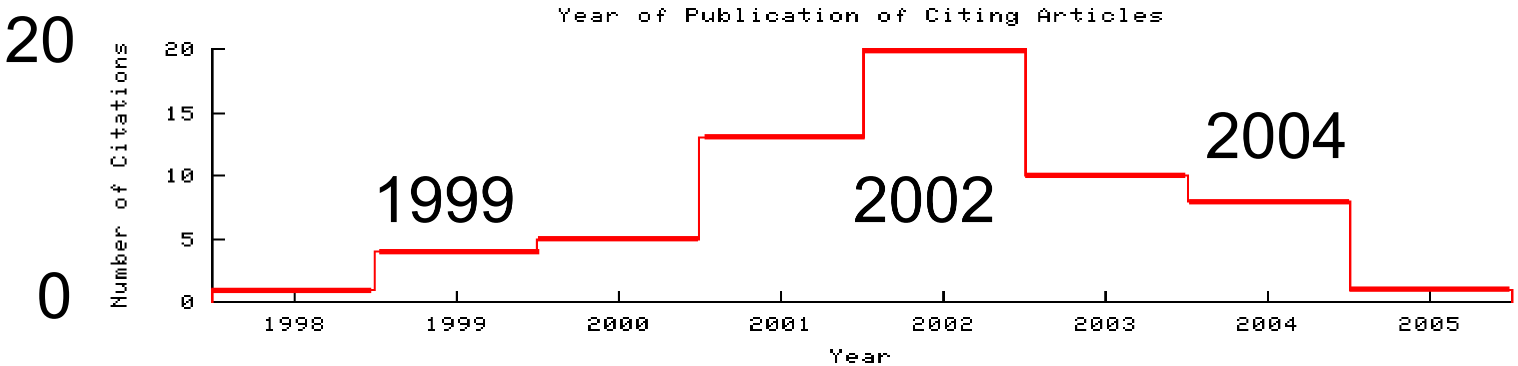
- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Neighborhood Data Sharing
- Future Work

Talk Outline

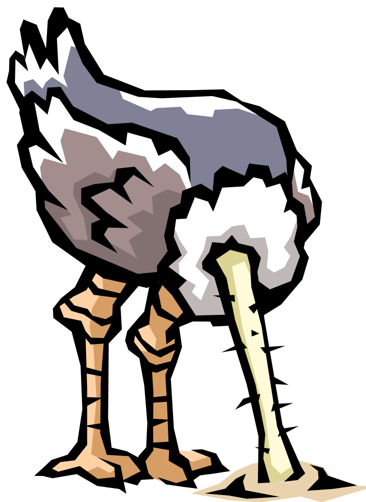
- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Neighborhood Data Sharing
- Future Work

Decreasing Interest in Energy Efficient Protocol Research

Citations for PAMAS paper by Year (from Citeseer)



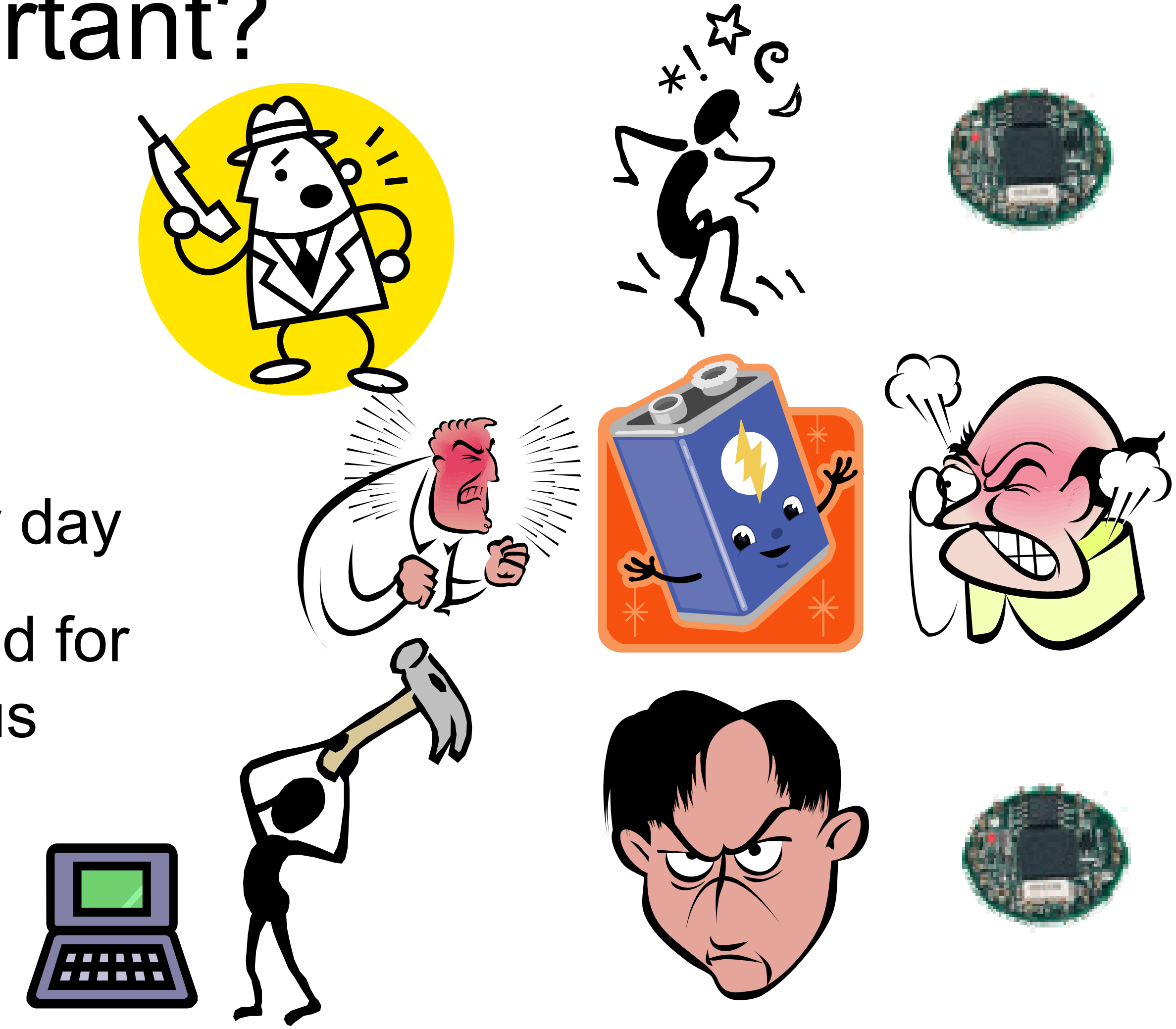
- Unfortunately, a significant portion of sensor and ad hoc network research ignores the issue
  - Promiscuous listening
  - Frequent “Hello” messages
  - Latency of network-wide flooding



Is Energy Efficiency Research Really Important?

YES!!!

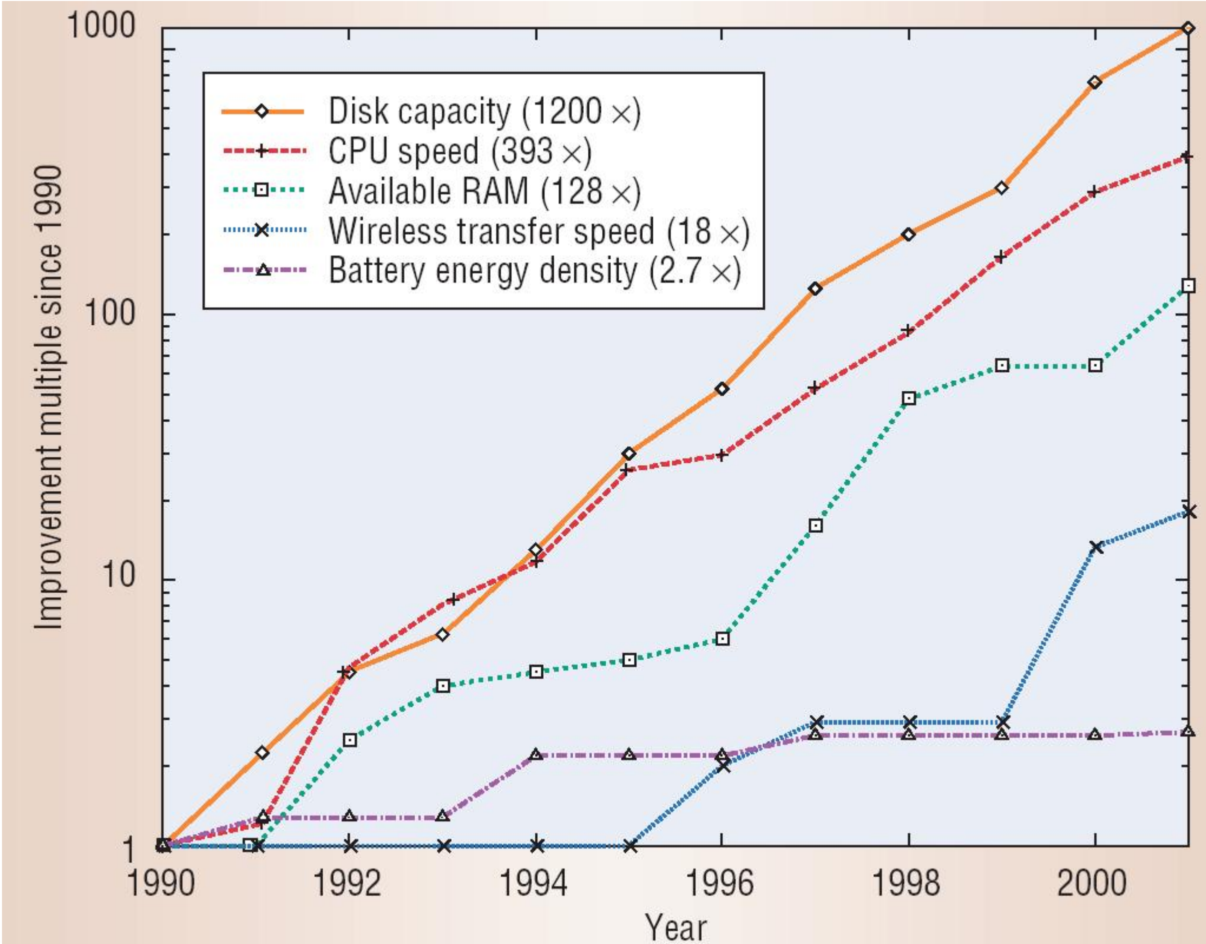
- ✓ It is a real world problem that affects wireless users every day
- ✓ Must be addressed for untethered ubiquitous wireless networks to become a reality



Won't Moore's Law Save Us?

NO!!!

Log Scale



1200 x
393 x
128 x
18 x
2.7 x

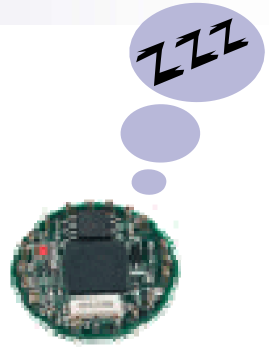
From "Thick Clients for Personal Wireless Devices" by Thad Starner in *IEEE Computer*, January 2002

Energy Consumption Breakdown

From Vodafone Symposium	Data Traffic (Laptop)	Voice Traffic (Cell Phone)
Display	45%	2%
TX	5%	24%
RX/Idle	10%	37%
CPU	40%	37%

- Solution spans multiple areas of research: networking, OS, architecture, and applications (e.g., GRACE project)
- Our work focuses on the networking component
- While applicable to laptops, our work is most beneficial to small/no display devices like sensors

How to Save Energy at the Wireless Interface

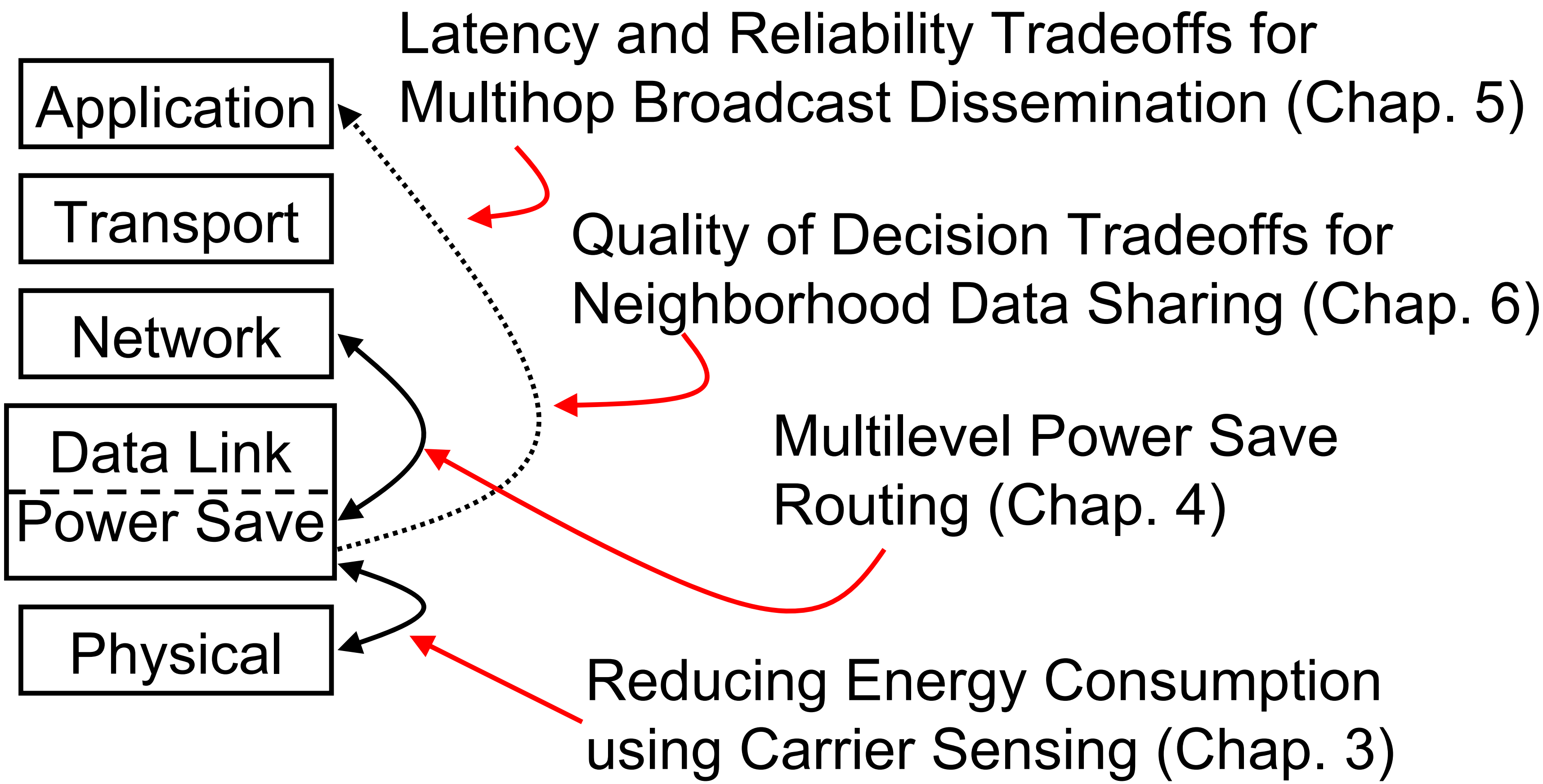


Specs for Mica2 Mote Radio

Radio Mode	**Power Consumption (mW)**
TX	81
RX/Idle	30
Sleep	0.003

- Sleep as much as possible!!!
- Fundamental Question: *When should a radio switch to sleep mode and for how long?*
  - Many similarities in power save protocols since all are variations of these two design decisions

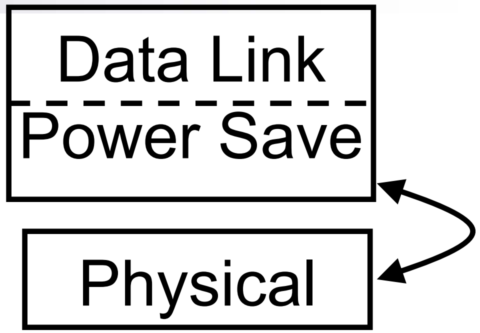
Our Contribution to Cross-Layer Design and Investigation

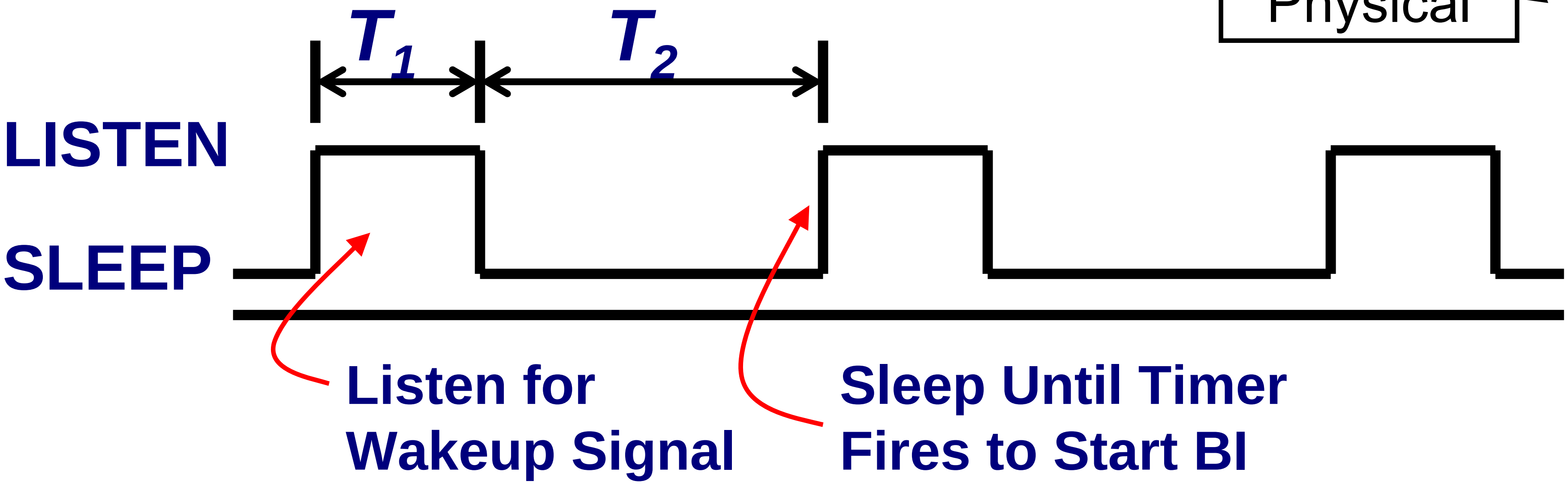


Talk Outline

- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Neighborhood Data Sharing
- Future Work

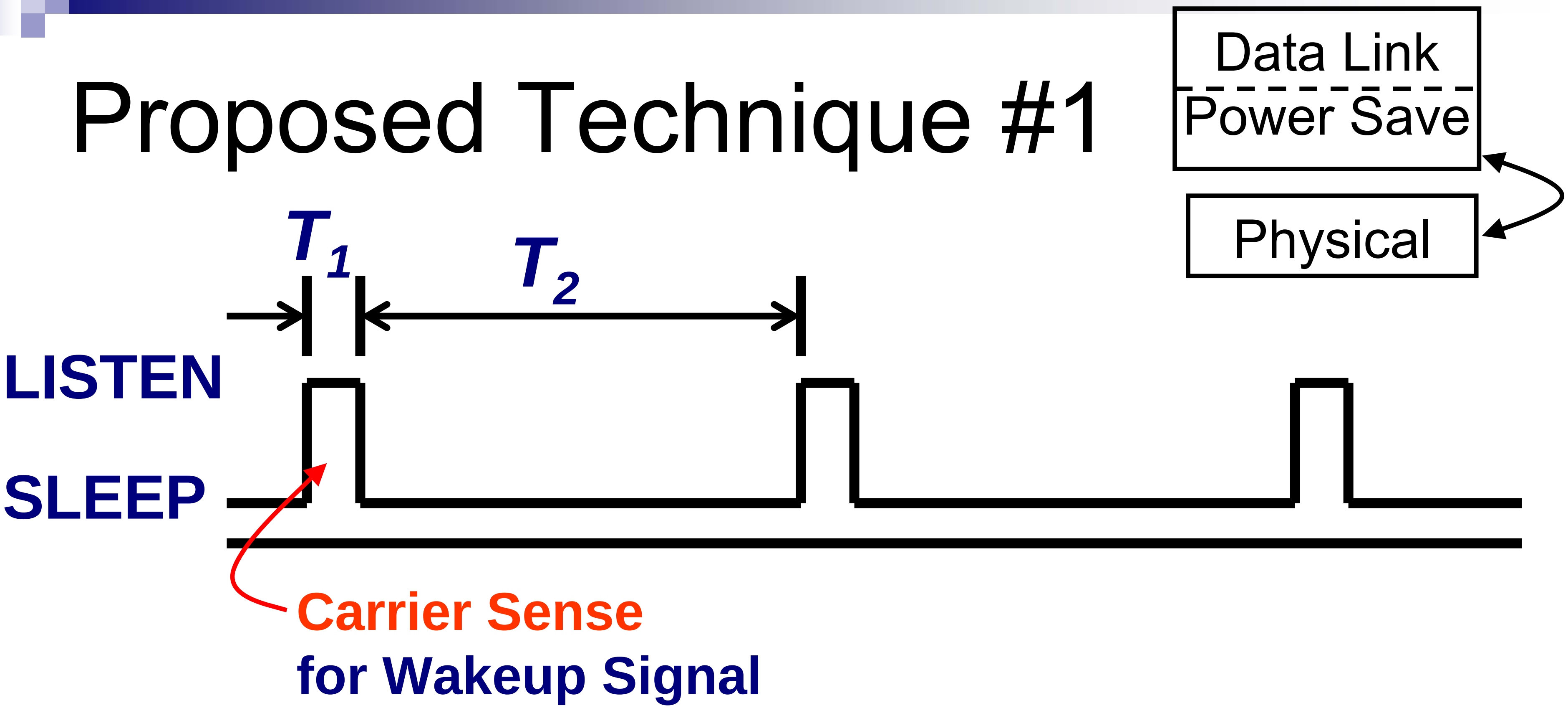
Common Design Used by Power Save Protocols





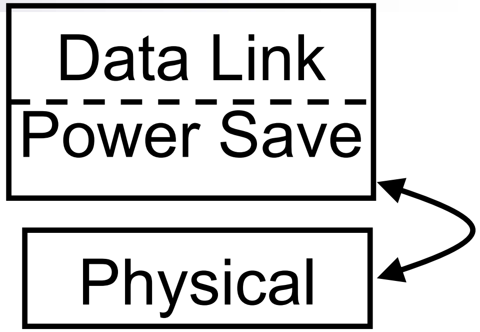
- $T_1 < T_2$
- Even with no traffic, node is awake for $T_1 / (T_1+T_2)$ fraction of the time
- T_1 is on the order of the time to receive a packet

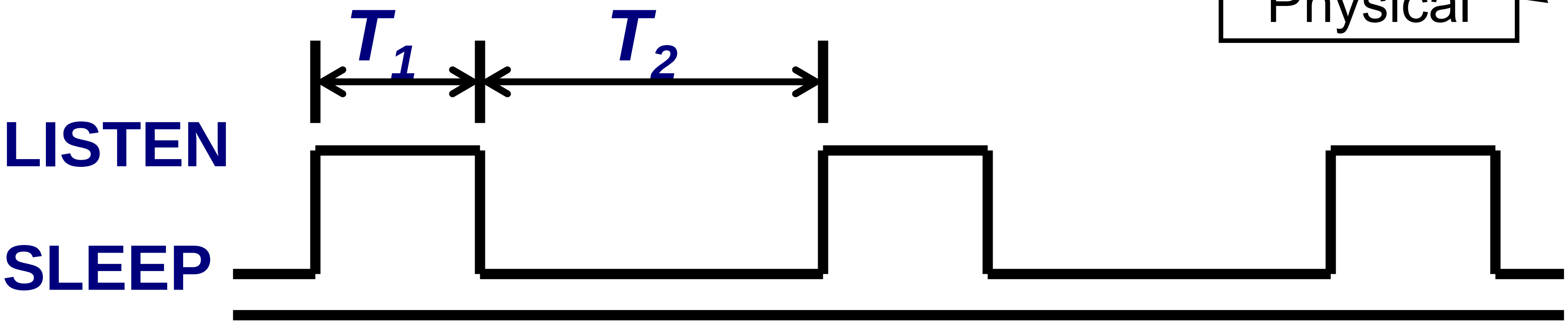
Proposed Technique #1



- Decrease T_1 using physical layer carrier sensing (CS)
- If carrier is sensed busy, then stay on to receive packet
- Typically, CS time << packet transmission time
  - E.g., 802.11 compliant hardware CS time ≤ 15 μs

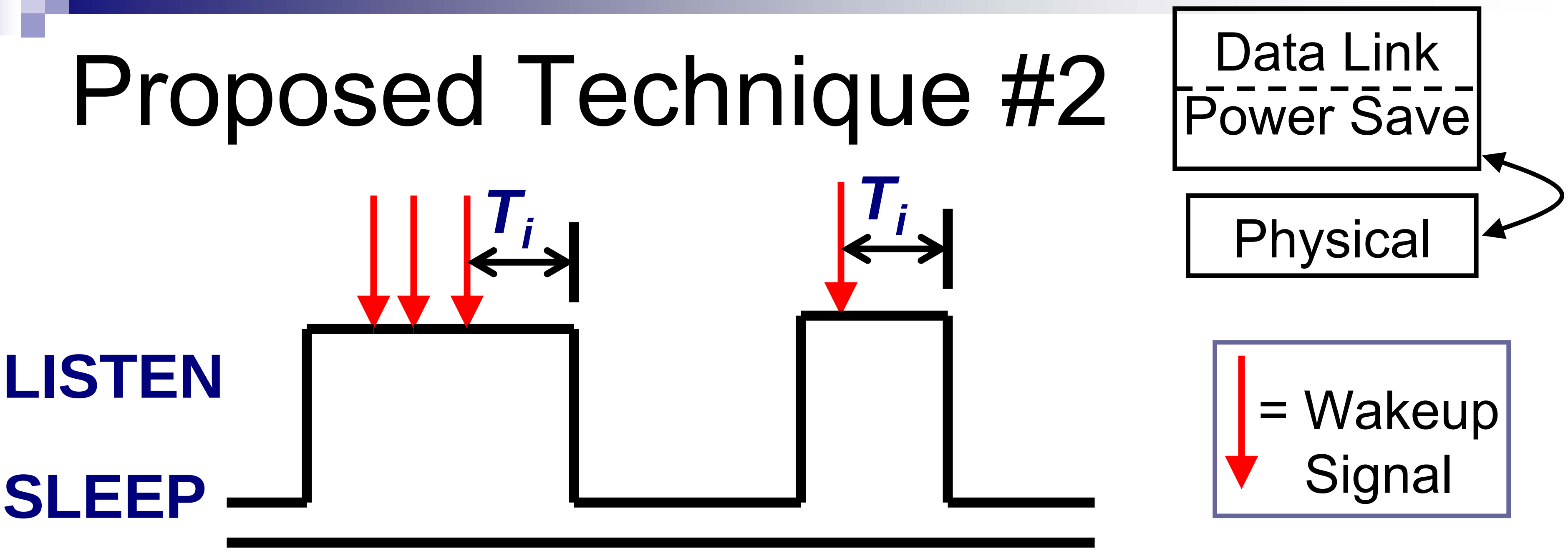
Another Observation





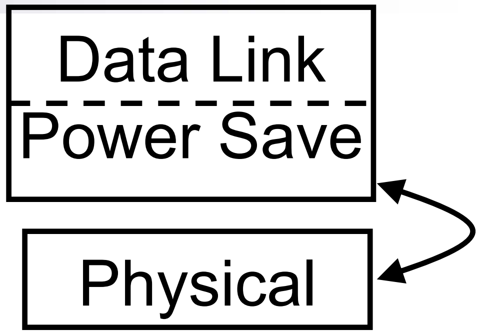
- T_1 is fixed regardless of how many wakeup signals are received
- Ideally, nodes stay on just long enough to receive all wakeup signals sent by their neighbors
  - If no signals are for them → return to sleep

Proposed Technique #2



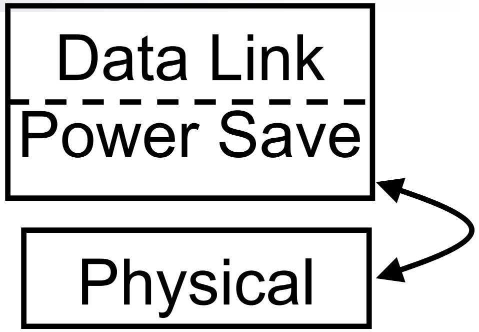
- Using physical layer CS, we dynamically extend the listening period for wakeup signals
- While previous work has proposed dynamic listening periods for 802.11 power save, ours is the first for single radio devices in multihop networks

Related Work



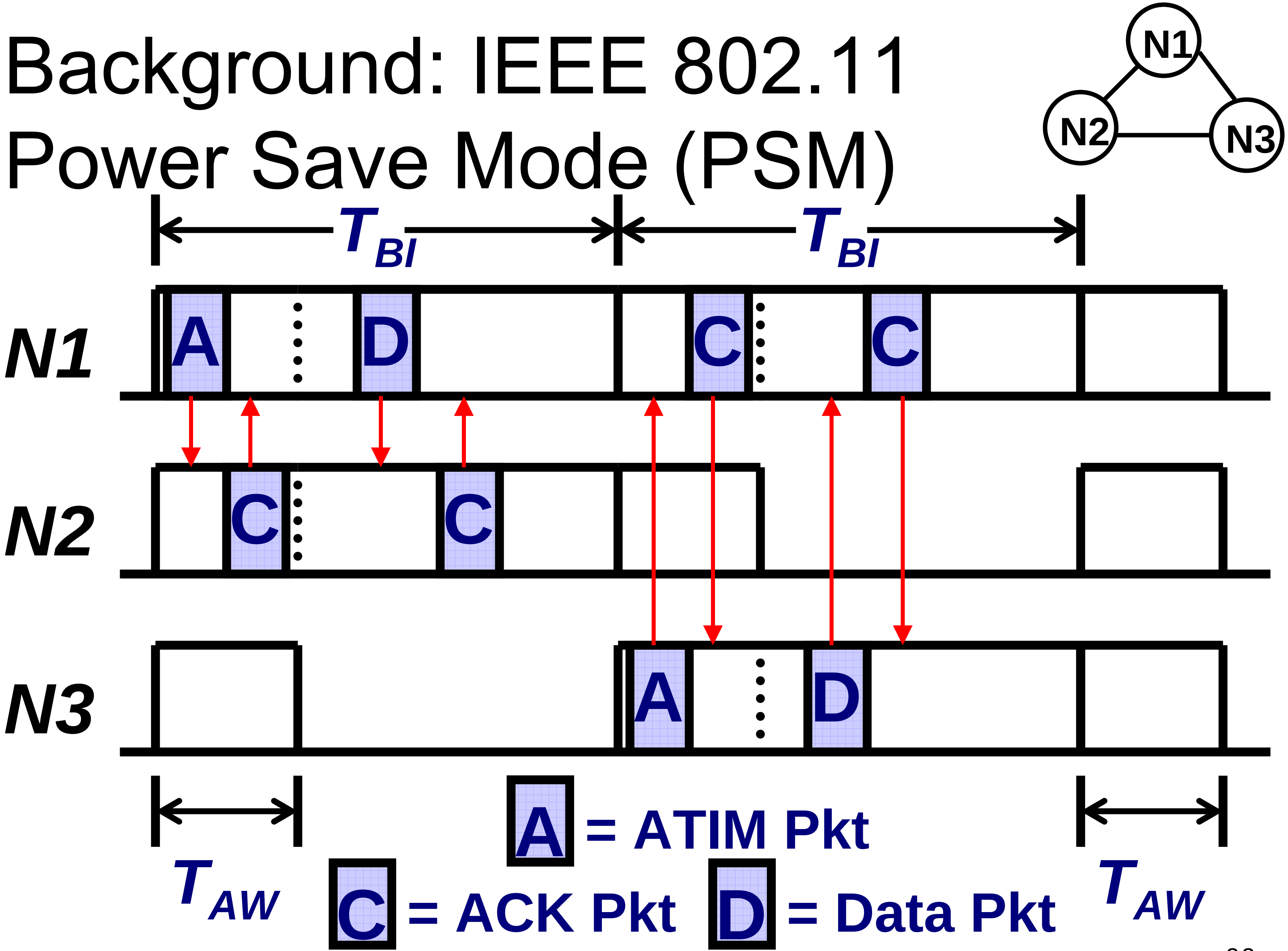
- Carrier Sensing (Concurrent Work)
  - B-MAC [Polastre04SenSys]: Make the packet preamble as large as the duty cycle
  - WiseMAC [ElHoiydi04Algosensors]: Send the packet preamble during the receiver’s next scheduled CS time
  - We apply CS to synchronous or out-of-band protocols
- Dynamic Listening Periods
  - T-MAC [VanDam03SenSys]: Extends S-MAC to increase the listen time as data packets are received
  - DPSM/IPSM [Jung02Infocom]: Extends 802.11 for dynamic ATIM windows in single-hop environments
  - We use physical layer CS to work in multihop environments without inducing extra packet overhead

Our Work

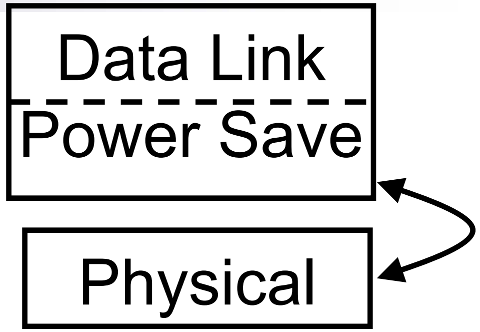


- We demonstrate how Technique #1 (Carrier Sensing for Signals) can be applied to two different types of power save protocols
- We show an application of Technique #2 (Dynamic Listening Period) can be combined with Technique #1 to create an energy efficient protocol

Background: IEEE 802.11 Power Save Mode (PSM)

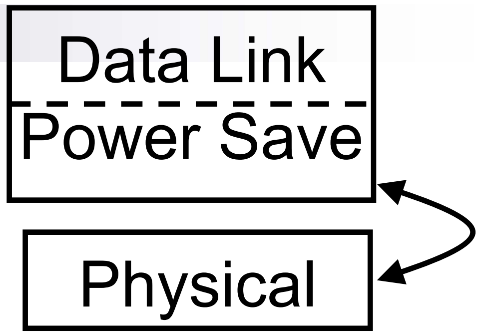


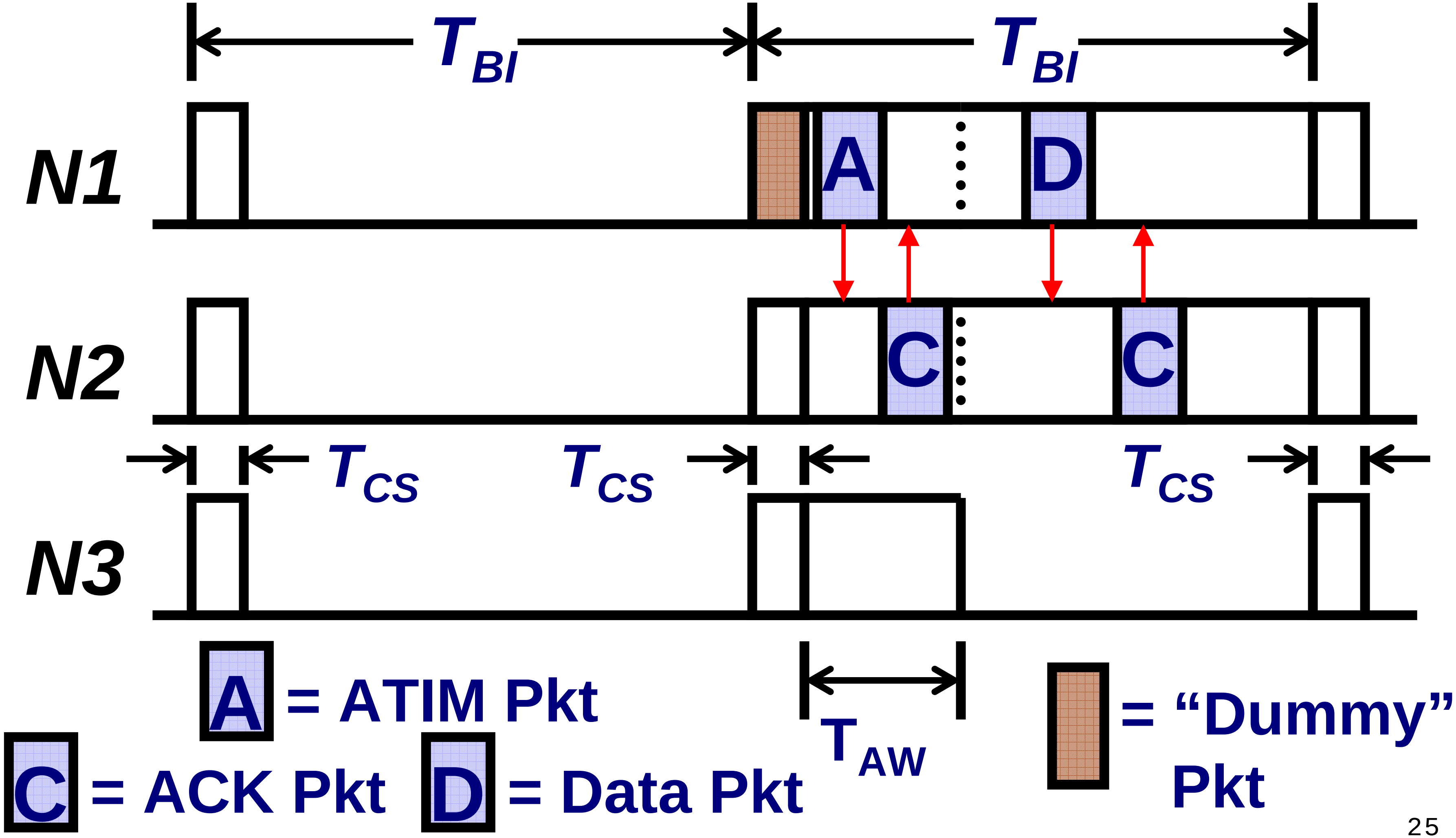
Background: IEEE 802.11 PSM



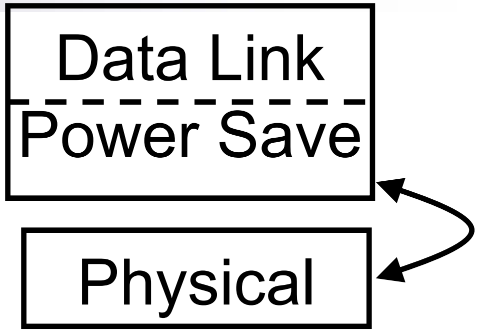
- Nodes are assumed to be synchronized
  - In our protocols, we assume that time synchronization is decoupled from 802.11 PSM
- Every beacon interval (BI), all nodes wake up for an ATIM window (AW)
- During the AW, nodes advertise any traffic that they have queued
- After the AW, nodes remain active if they expect to send or receive data based on advertisements; otherwise nodes return to sleep until the next BI

Applying Technique #1 to 802.11 PSM



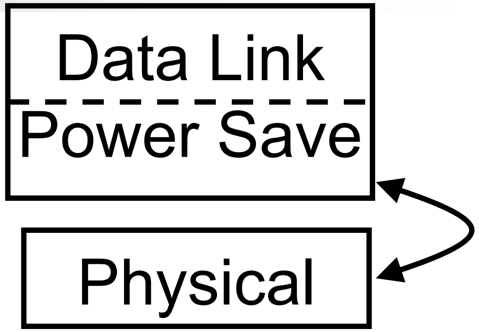


Applying Technique #1 to 802.11 PSM



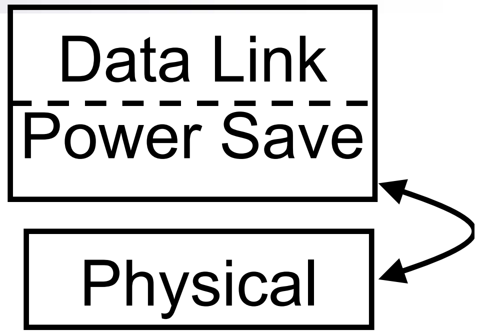
- Each beacon interval, nodes carrier sense the channel for T_{CS} time, where $T_{CS} << T_{AW}$
- If the channel is carrier sensed busy, nodes remain on for the remainder of the AW and follow the standard 802.11 PSM protocol
- If the channel is carrier sensed idle, nodes return to sleep without listening during the AW
- Node with data to send transmits a short "dummy" packet during T_{CS} to signal neighbors to remain on for AW

Observations

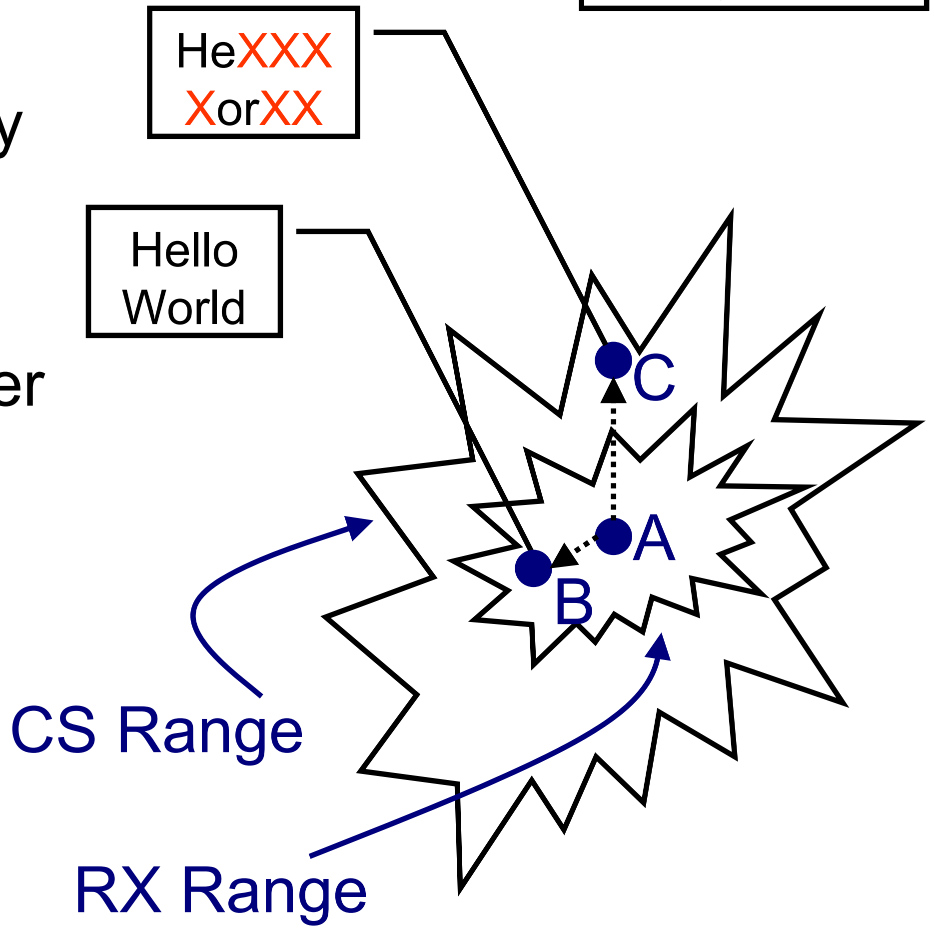


- When there are no packets to be advertised, nodes use significantly less energy
- Average latency is slightly longer
  - Packets that arrive during the AW are advertised in 802.11 PSM, but may not be with our technique
  - First packet cannot be sent until $T_{CS}+T_{AW}$ after beginning of BI instead of just T_{AW}
- False positives may occur when nodes carrier sense the channel busy due to interference
- Can be adapted to other types of power save protocols (e.g., TDMA)

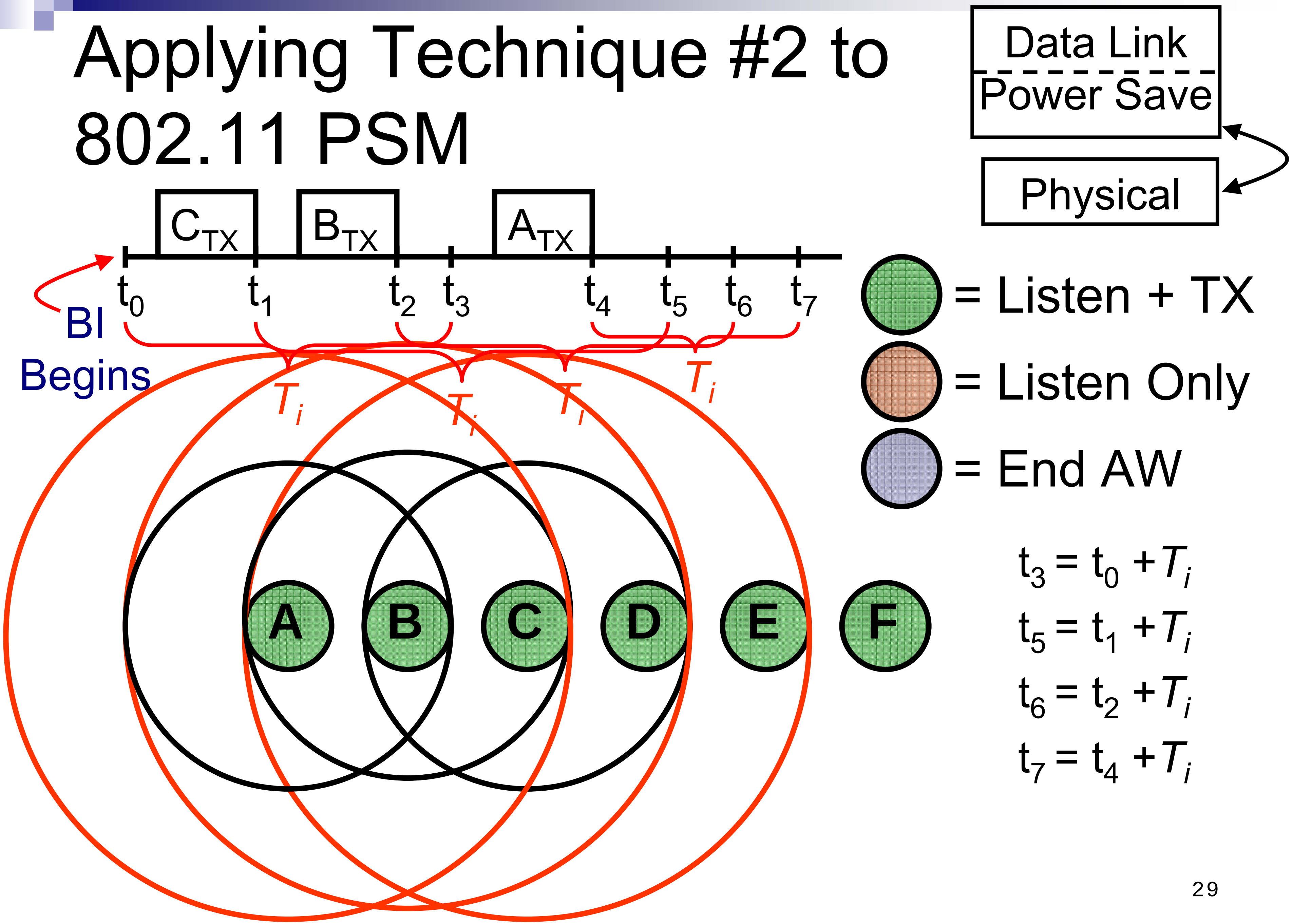
Background: RX Threshold vs. CS Threshold



- RX Threshold: received signal strength necessary for a packet to be correctly received
- CS Threshold: received signal strength to consider the channel busy
- We assume that usually CS range ≥ 2*RX range
  - If this is not true, our technique gracefully degrades to a fixed AW scheme



Applying Technique #2 to 802.11 PSM



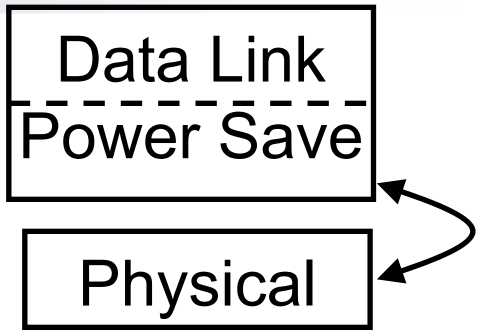
$t_3 = t_0 + T_i$

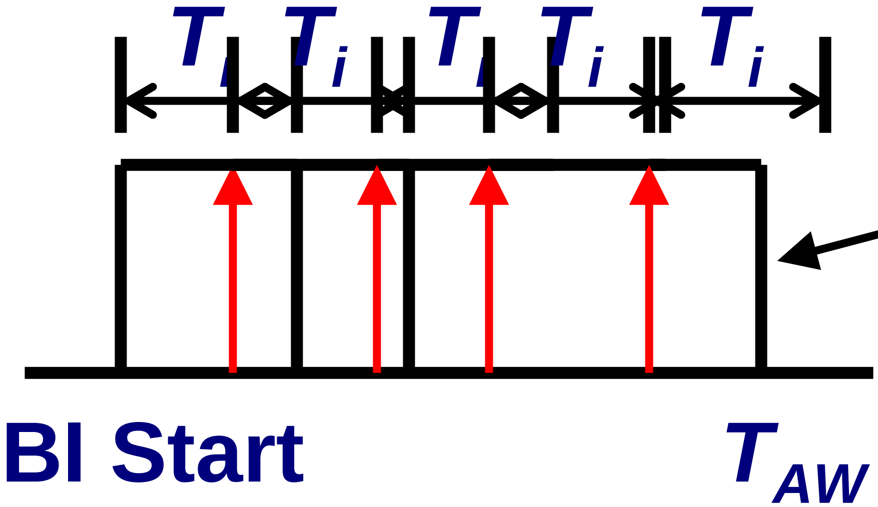
$t_5 = t_1 + T_i$

$t_6 = t_2 + T_i$

$t_7 = t_4 + T_i$

Applying Technique #2 to 802.11 PSM: Listening

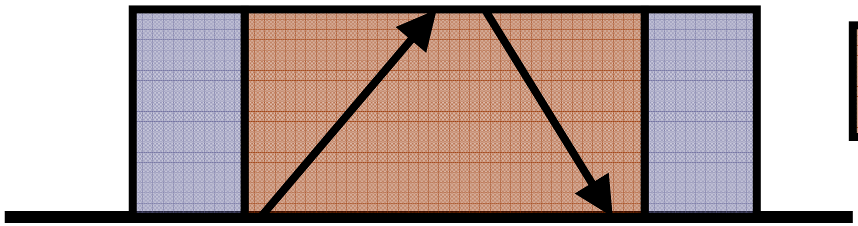




Sleep according to 802.11 PSM rules

= TX, RX, or CS busy event

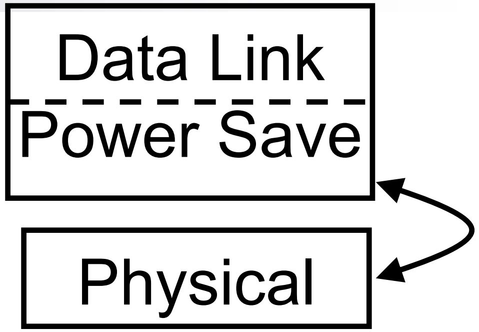
T_i =



= Max Contention Time

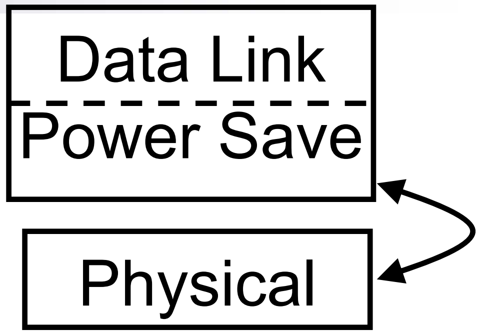
= ATIM/ATIM-ACK Handshake Time

Applying Technique #2 to 802.11 PSM: Listening



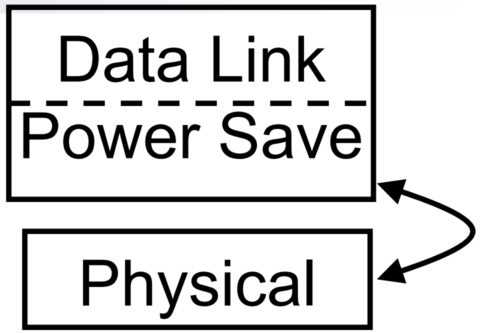
- At the beginning of each BI, listen for T_i time ($T_{CS} < T_i < T_{AW}$)
- When a packet is sent or received OR the channel is carrier sensed busy, extend listening time by T_i
- Set maximum on how long the listening time can be extended since the beginning of the BI

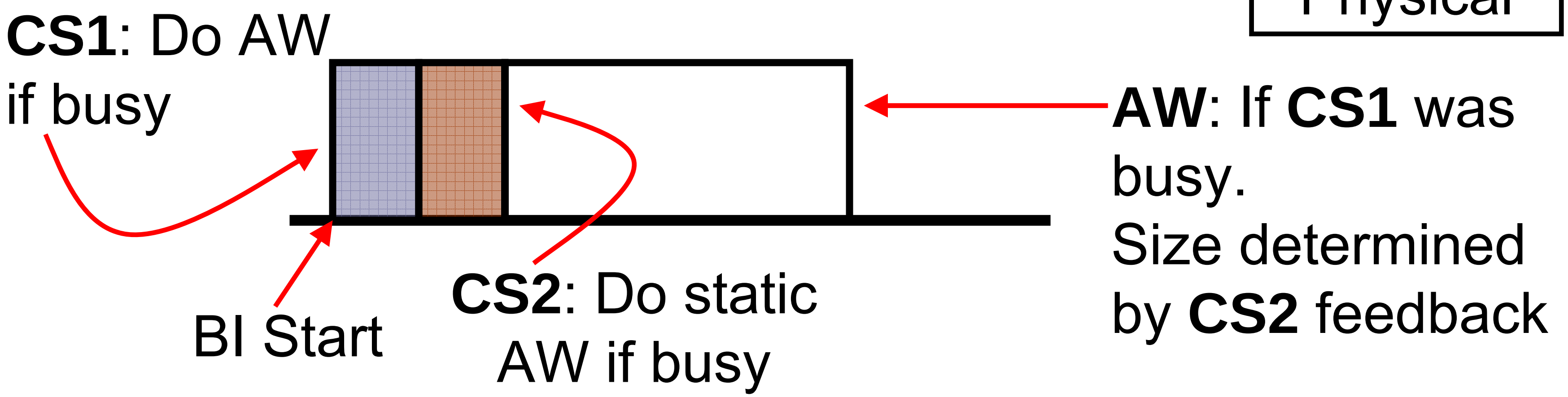
Applying Technique #2 to 802.11 PSM: Sending



- Node with packets to advertise
  - If a packet has been received above the RX Threshold within T_i time, all neighbors are assumed to be listening
  - Otherwise, the node conservatively assumes that its intended receiver(s) is sleeping and waits until the next beacon interval to advertise the packet
- T_i is set such that a sender can lose one MAC contention and its receiver will continue listening

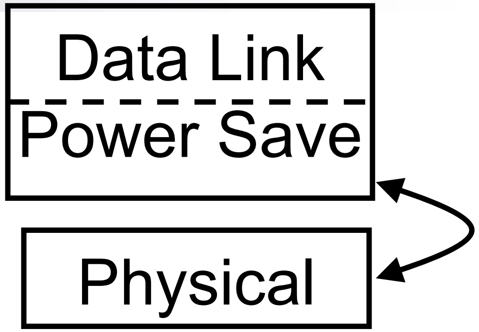
Combining Technique #1 and Technique #2



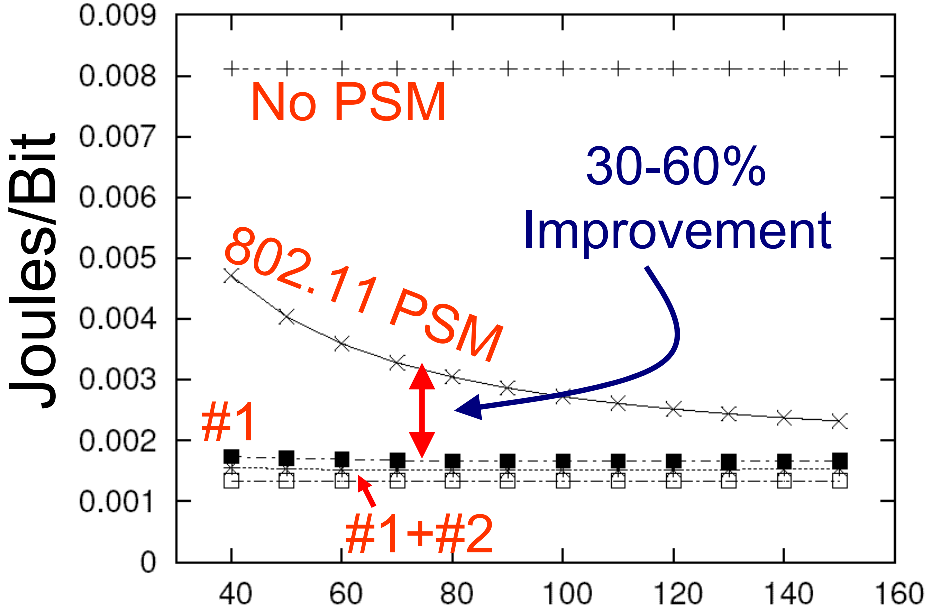


- First CS period indicates whether an AW is necessary
- Second CS period indicates whether AW size should be fixed or dynamic according to Technique #2
  - If a sender repeatedly fails using a dynamic AW, this is a fallback to the original protocol

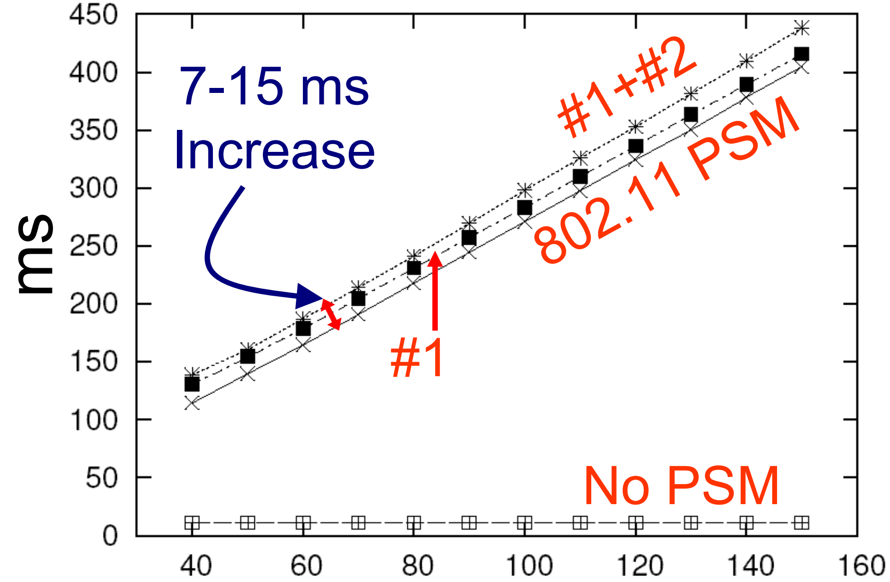
Summary of Results



Energy



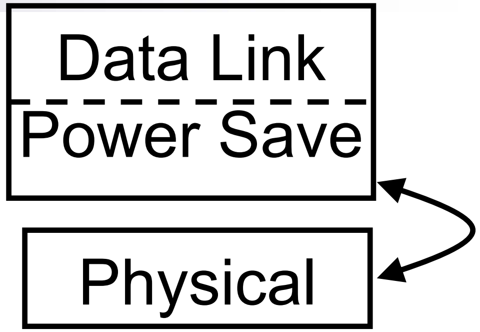
Latency



Beacon Interval (ms), AW = 20 ms

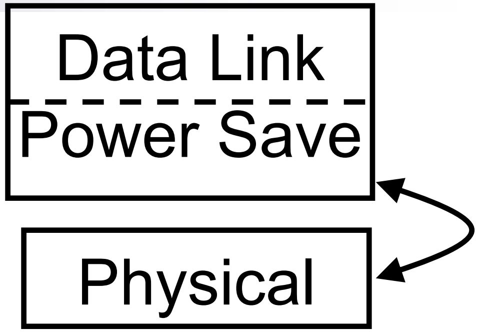
Latency Increase: (1) Additional CS periods, (2) Packets arriving during AW, (3) For Technique #2, postponed advertisements

Application to Other Power Save Protocols



- Out-of-band power save protocols use an external mechanism for wakeup signaling
- Our thesis also presents the application of Technique #1 to an out-of-band (OOB) power save protocol (Section 3.2)
- Analysis and simulation show significant gains when OOB protocol does not already use some form of carrier sensing

Summary

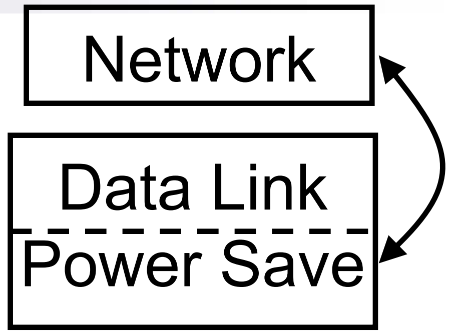


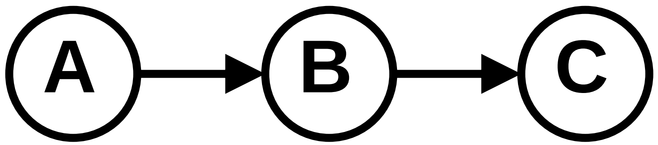
- Application of physical layer CS to synchronous power save protocol to reduce listening interval
- Physical layer CS for dynamic listening interval for single radio devices in multihop networks
- Application of physical layer CS to further improve OOB power save protocol

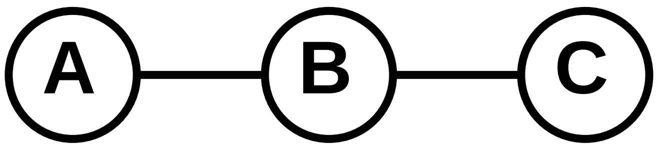
Talk Outline

- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Neighborhood Data Sharing
- Future Work

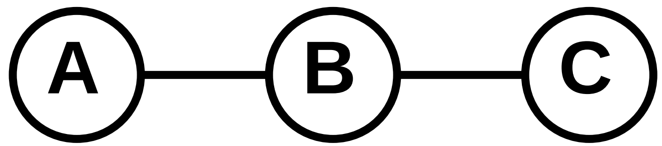
Utility of Cross-Layer Design at the Network Layer





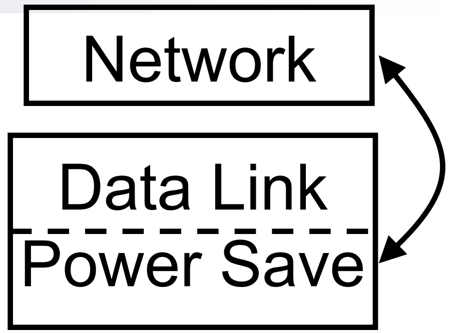


Isolated Power Save:
A—B and B—C make decisions independently



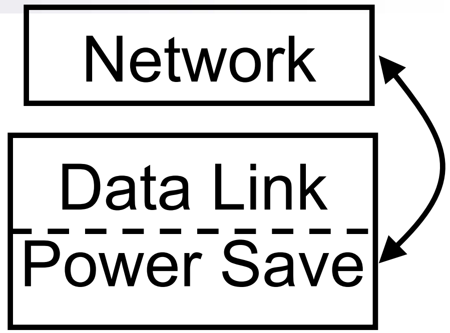
Cross-Layer Power Save:
A—B and B—C can coordinate decisions

Related Work: Cross-Layer Power Save Routing



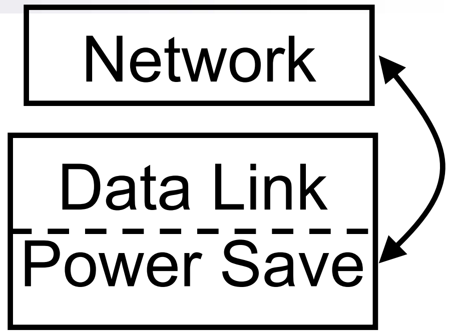
- ODPM [Zheng03Infocom]: Nodes on an active route turn off power save while all other nodes use 802.11 PSM
- TITAN [Sengul05MC2R]: Extends ODPM; route discovery modified to favor routes that are already active
- Route discovery is limited to two choices:
  - Low latency, high energy paths
  - High latency, low energy paths
- Our work exposes a wider range of energy-latency tradeoffs during route discovery

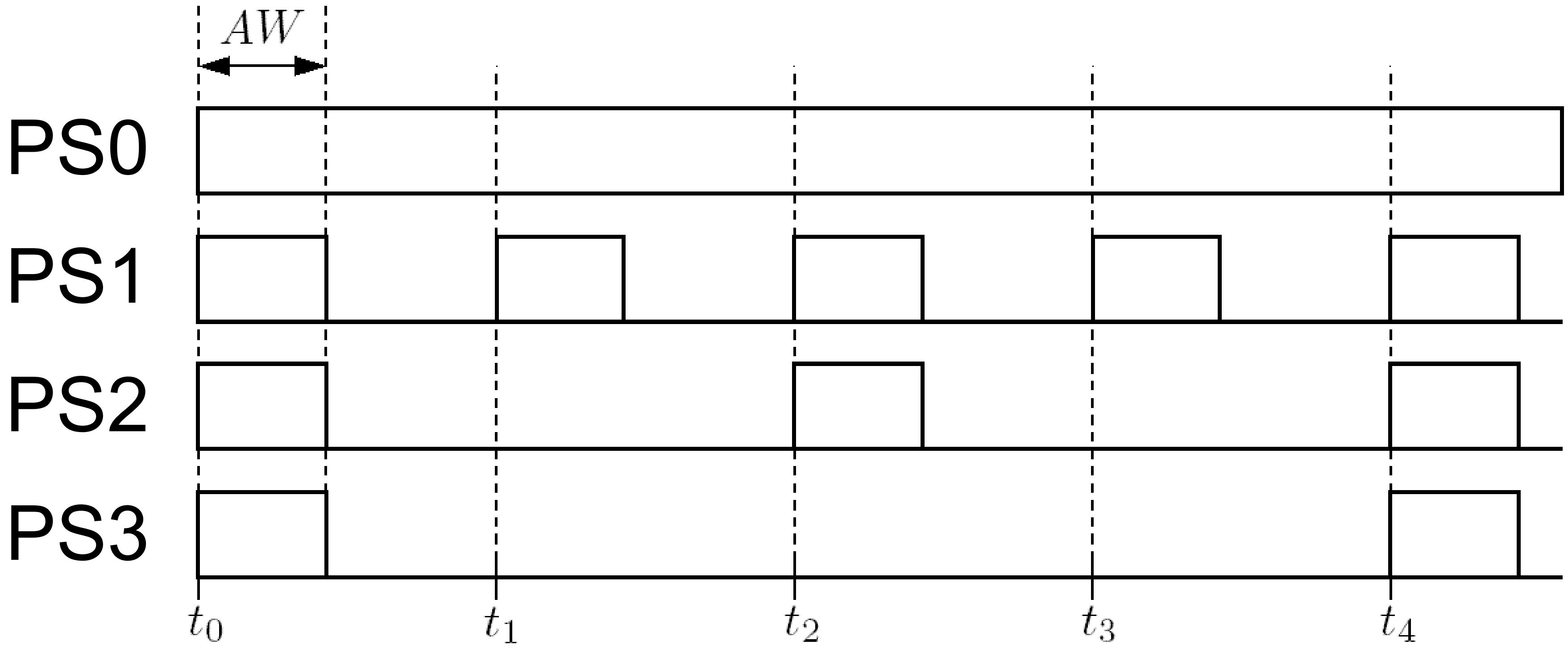
Our Work



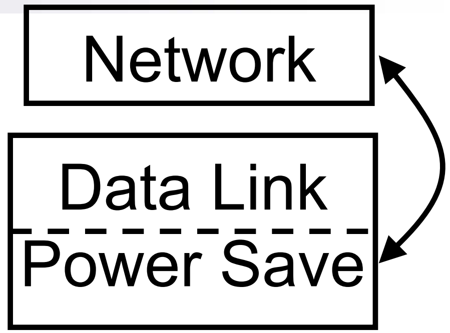
- Routing protocols designed for nodes using multiple levels of power save
  - Protocol to discover paths with acceptable power save-induced latency while reducing energy consumption
  - Source-to-sink routing protocol to reduce latency while increasing network lifetime

Multilevel Power Save: 802.11 PSM Example



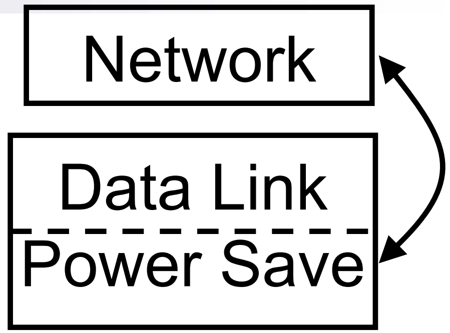


Multilevel Power Save



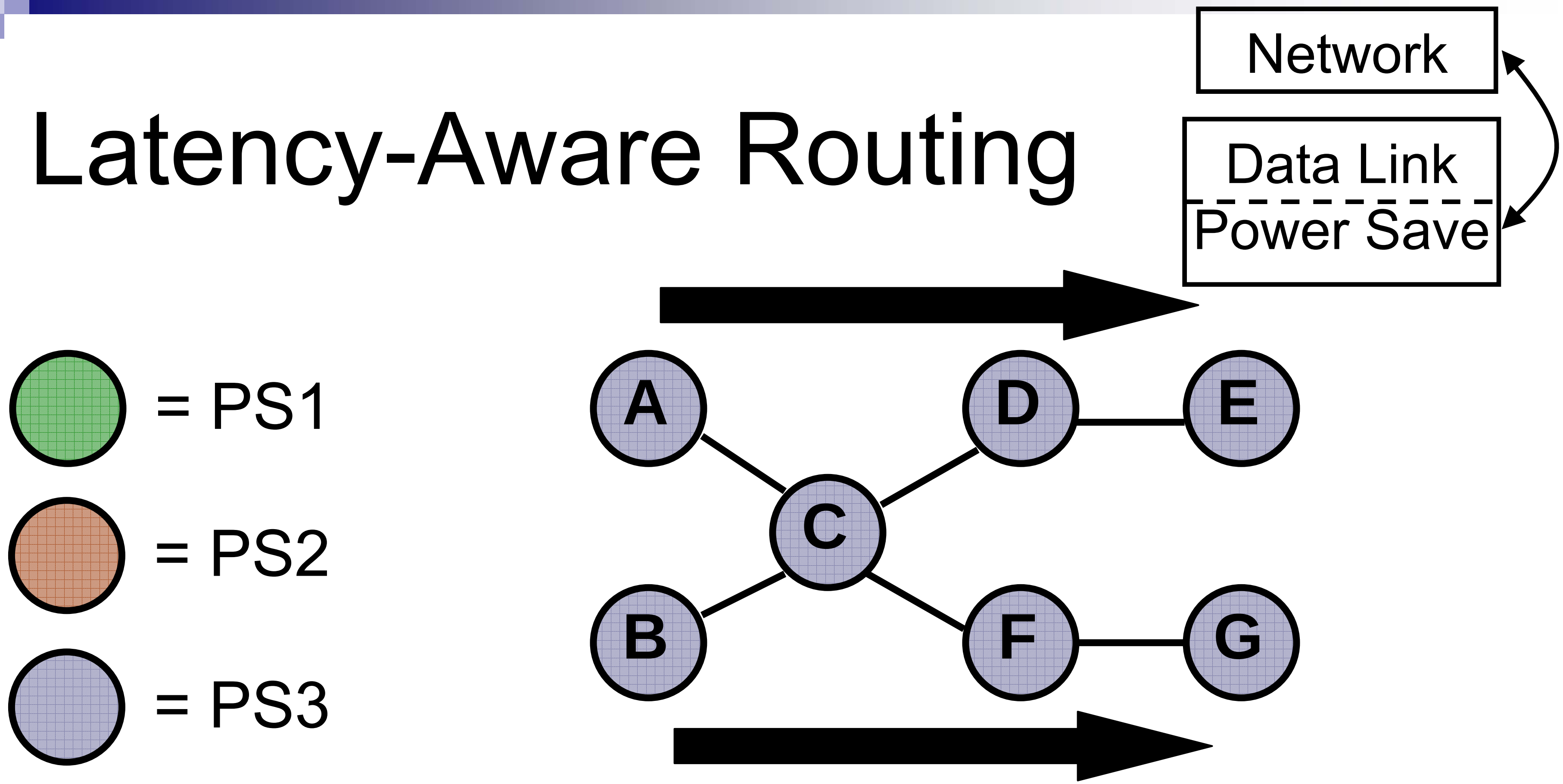
- Each level presents a different energy-latency tradeoff (i.e., higher energy → lower latency)
- 802.11 PSM
  - Nodes are synchronized to a reference point
  - T_{BI} for i-th power level: $T_{BI}(i) = 2^{i-1} * BI_{base}$
    - $i > 0$ and $T_{BI}(1) = BI_{base}$
- Other PS protocols such S-MAC and WiseMAC can be modified similarly

Latency-Aware Routing



- Goal: Create path:
  1. With end-to-end *power save induced* latency less than *L*
  2. That requires the lowest increase in energy consumption for nodes on the path
- *L* can be given by application requesting path
- Route replies include the power save state of each node on a path to allow the source to calculate the end-to-end latency
- Choose lowest cost (e.g., hop count) path if routes exist with a latency less than *L*

Latency-Aware Routing

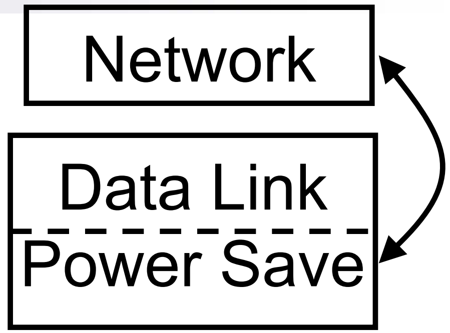


PS1 > PS2 > PS3

Where “X > Y” means X
has higher energy and lower latency than Y

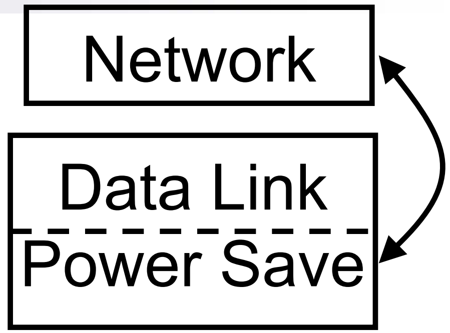
A—E requires lower latency than B—G

Latency-Aware Routing



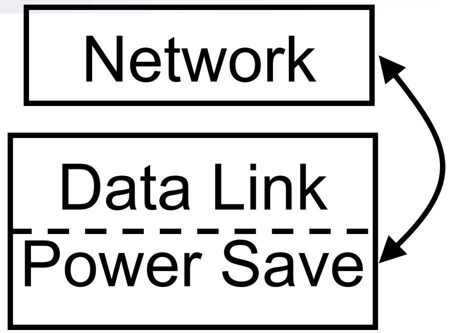
- If no path with latency less than *L* exists, choose the path that requires the smallest increase in energy consumption and send a packet with the PS states for the route
- Piggyback these PS states on every data packet sent along the path
- Nodes remain in lowest energy PS state that maintains an acceptable latency for *all* flows traveling through it

Network Lifetime-Aware Routing



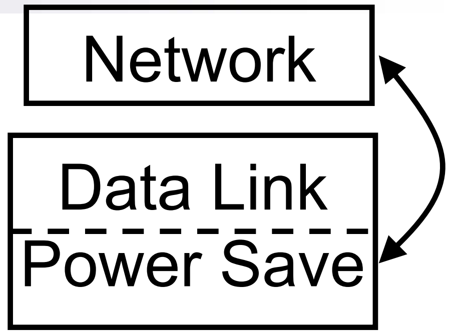
- Designed for source-to-sink routing (e.g., sensor and hybrid networks)
- Goal: Increase network lifetime while reducing latency despite asymmetric per node loads
- Based on observation that nodes closer to the sink use more energy forwarding packets

Network Lifetime-Aware Routing



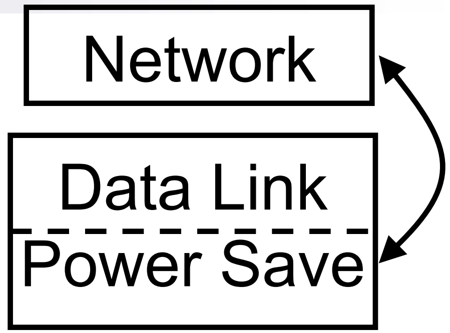
- Beacon intervals are length T_{bi}
- Nodes use T_w time each interval listening for wakeup signals
- Nodes use T_f time per interval forwarding packets (i.e., TX, RX, MAC contention)
- Fraction of time spent in non-sleep mode, $F_{ns} = (1/T_{bi}) * (T_w + T_f)$
- Latency = sum of T_{bi}'s at each hop on path

Network Lifetime-Aware Routing



- $F_{ns} = (1/T_{bi}) * (T_w + T_f)$
  Latency = sum of T_{bi}'s at each hop on path
  - Node lifetime varies inversely with F_{ns}
  - T_w fixed for all nodes
  - T_f is greater for nodes closer to sink
- Adjust T_{bi} per node such that F_{ns} is constant for all nodes on a path
- ***Thus, all nodes have the same lifetime and nodes farther from the sink reduce the end-to-end latency with shorter duty cycles***

Summary

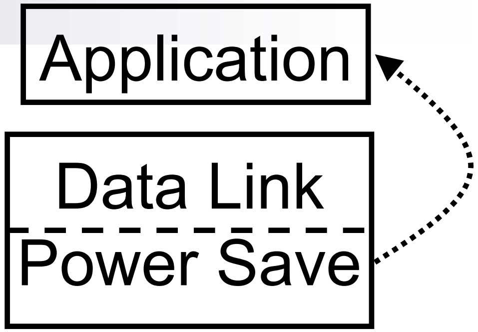


- Proposed the concept of multilevel power save as a cross-layer design technique between the link layer and network layer
- Introduced routing protocol with fine-grain control for creating paths with acceptable latency while reducing energy consumption
- Proposed routing protocol to balance energy consumption while reducing latency in source-to-sink networks where the load is unbalanced
- *Future Work*
  - Simulate and evaluate both routing protocols
  - Integrate multilevel routing with physical layer carrier sensing

Talk Outline

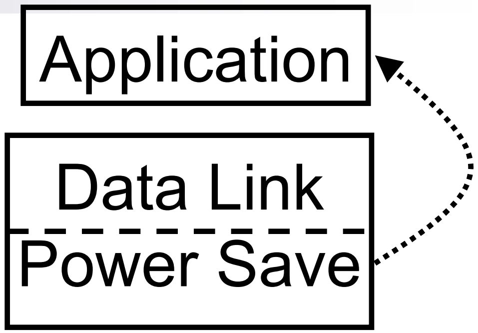
- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Neighborhood Data Sharing
- Future Work

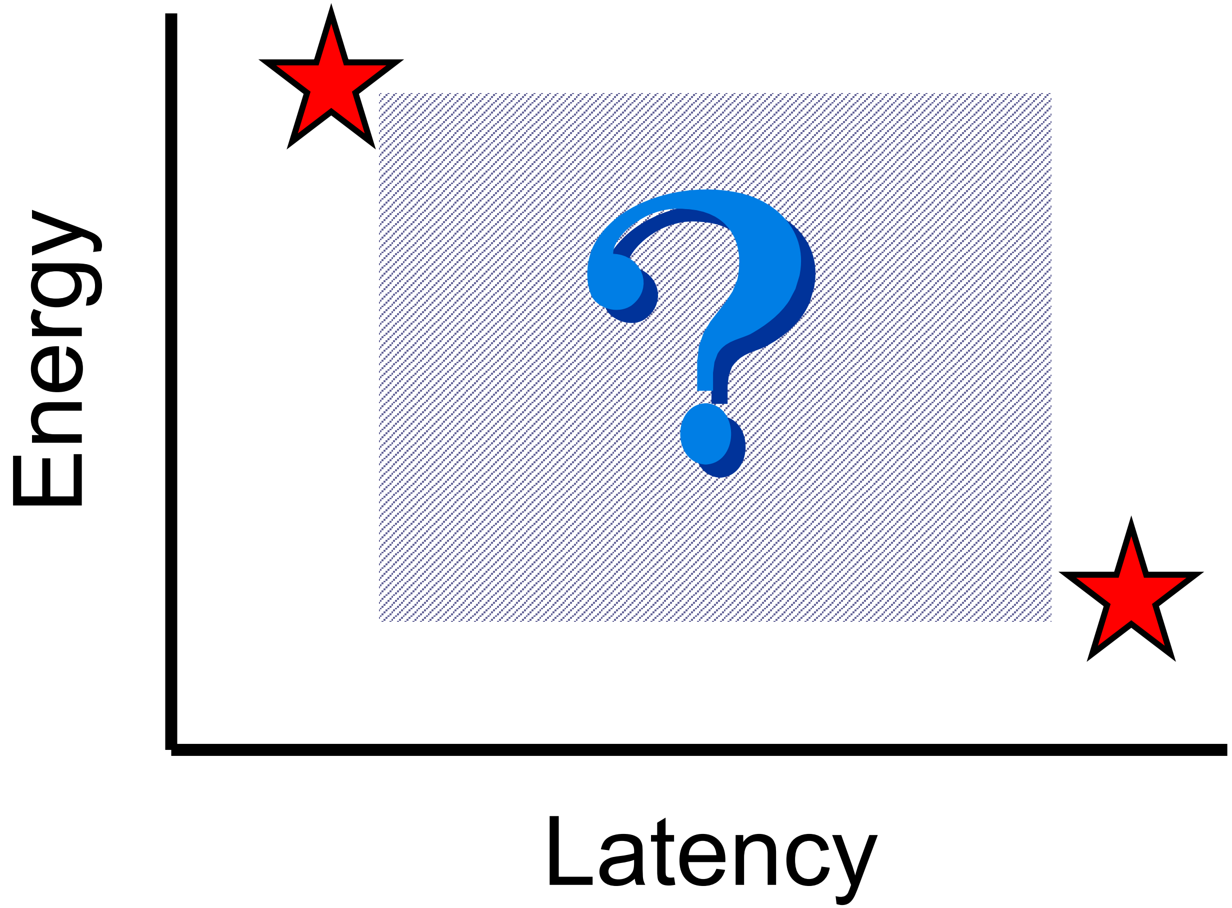
Multihop Broadcast Applications



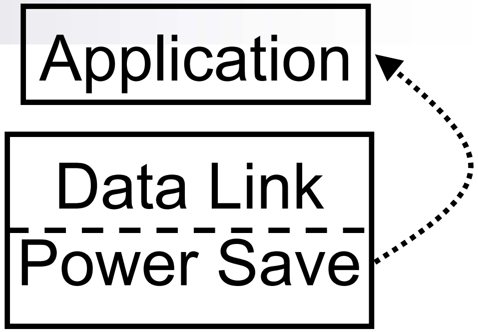
- Broadcast is a common means of disseminating and querying data in multihop wireless networks
- Example Applications
  - On-demand route discovery
  - Code distribution
  - Querying for sensor data
- What cross-layer effects arise in such applications as a result of power save?
  - Latency
  - Reliability

Energy-Latency Options



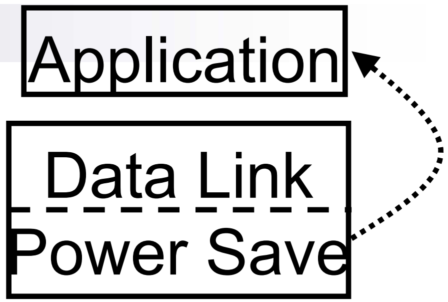


Our Work



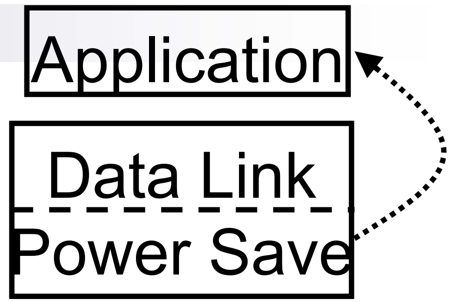
- Design a protocol that gives network administrators control over the energy-latency tradeoff for multihop broadcast applications
- Characterize the achievable latency and reliability performance for such applications that results from using power save protocols

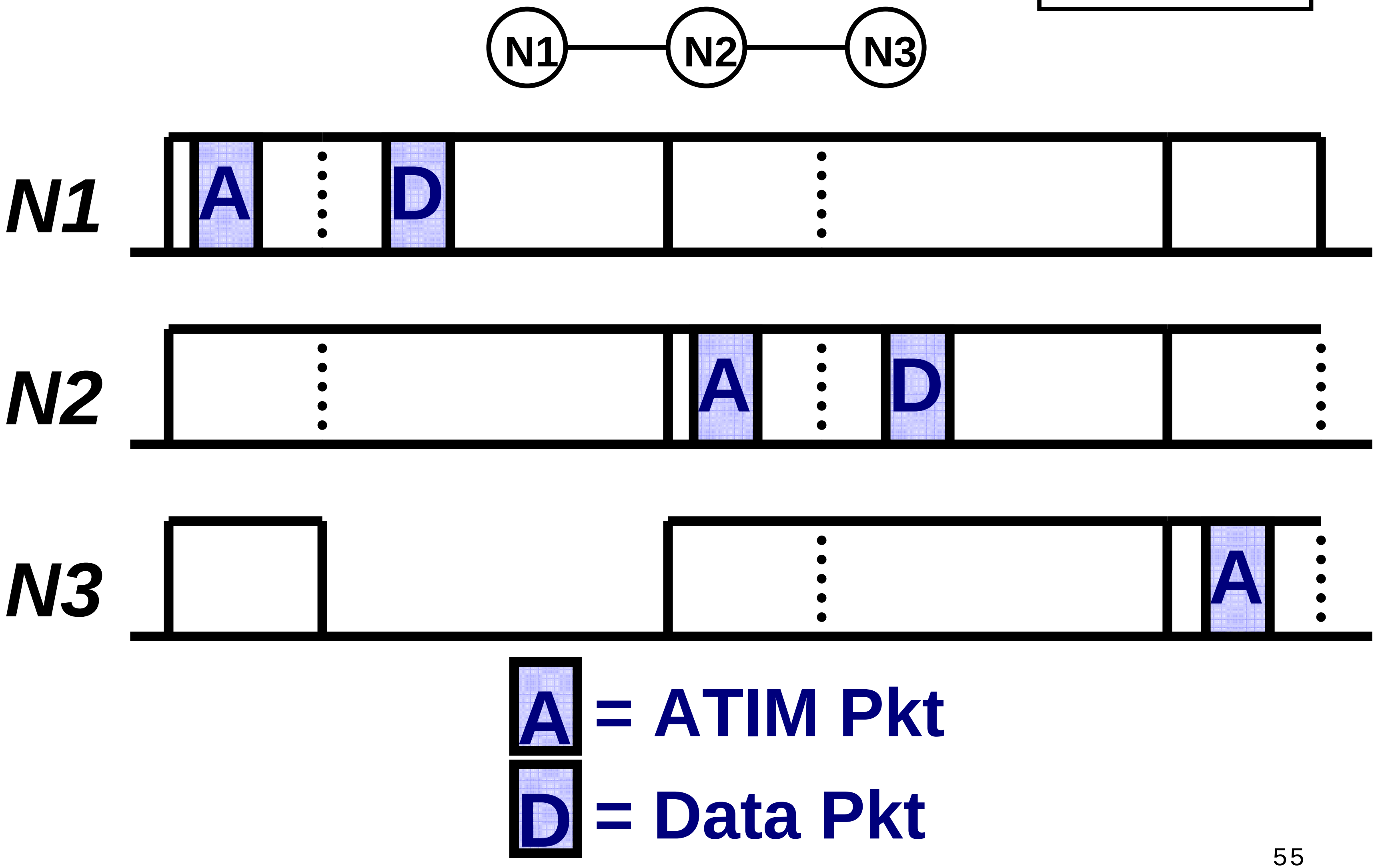
Sleep Scheduling Protocols



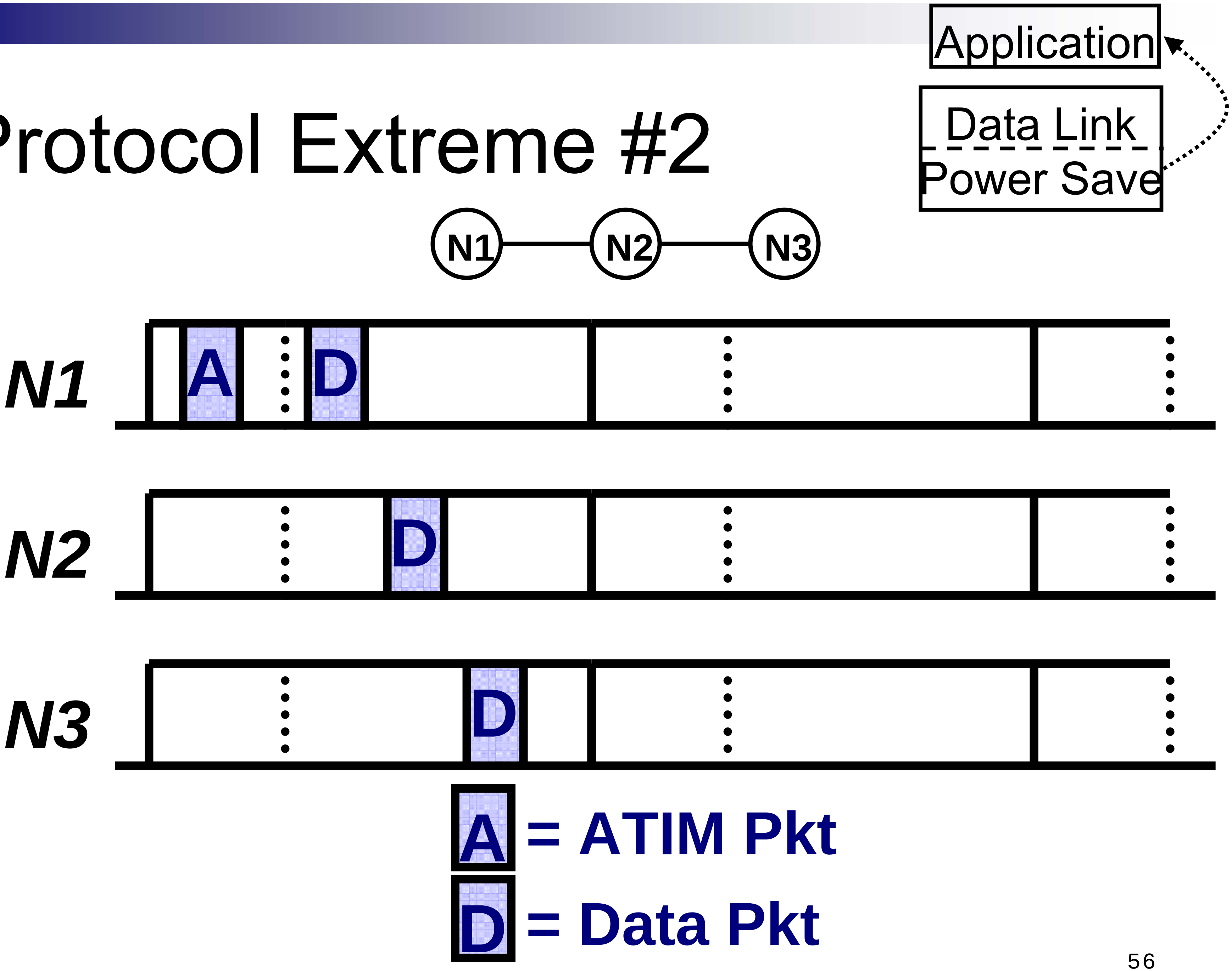
- Nodes have two states: active and sleep
- At any given time, some nodes are active to communicate data while others sleep to conserve energy
- Examples
  - IEEE 802.11 Power Save Mode (PSM)
    - Most complete and supports broadcast
    - Not necessarily directly applicable to sensors
  - S-MAC/T-MAC
  - STEM

Protocol Extreme #1

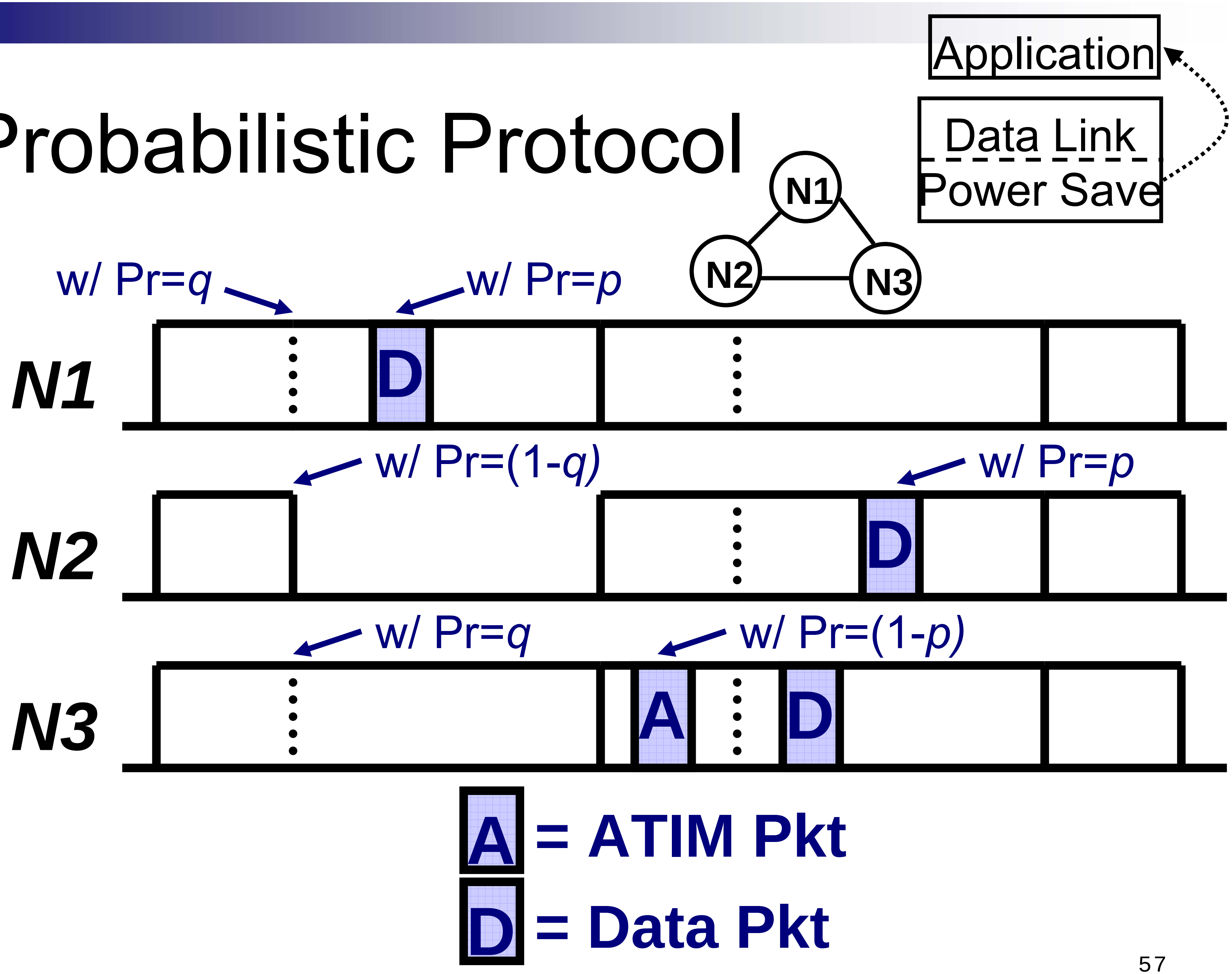




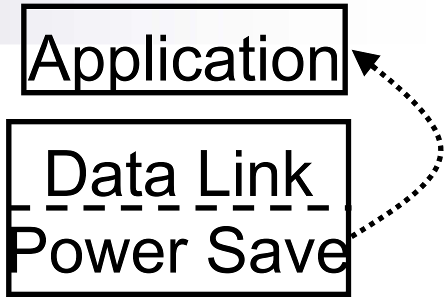
Protocol Extreme #2



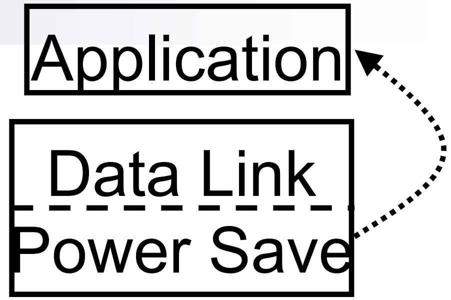
Probabilistic Protocol



Probability-Based Broadcast Forwarding (PBBF)



- Introduce two parameters to sleep scheduling protocols: *p* and *q*
- When a node is scheduled to sleep, it will remain active with probability *q*
- When a node receives a broadcast, it sends it immediately with probability *p*
  - With probability (*1-p*), the node will wait and advertise the packet during the next AW before rebroadcasting the packet

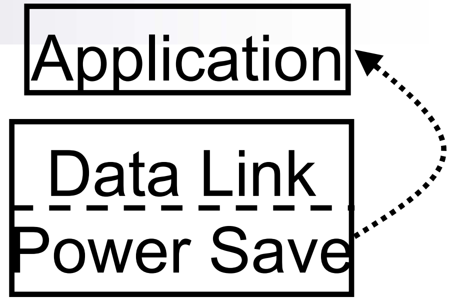


Observations

- p=0, q=0 equivalent to the original sleep scheduling protocol
- p=1, q=1 approximates the “always on” protocol
  - Still have the ATIM window overhead

- Effects of p and q on metrics:

	Energy	Latency	Reliability
$p\uparrow$	---	$\downarrow$ if $q > 0$	$\downarrow$ if $q < 1$
$q\uparrow$	$\uparrow$	$\downarrow$ if $p > 0$	$\uparrow$ if $p > 0$

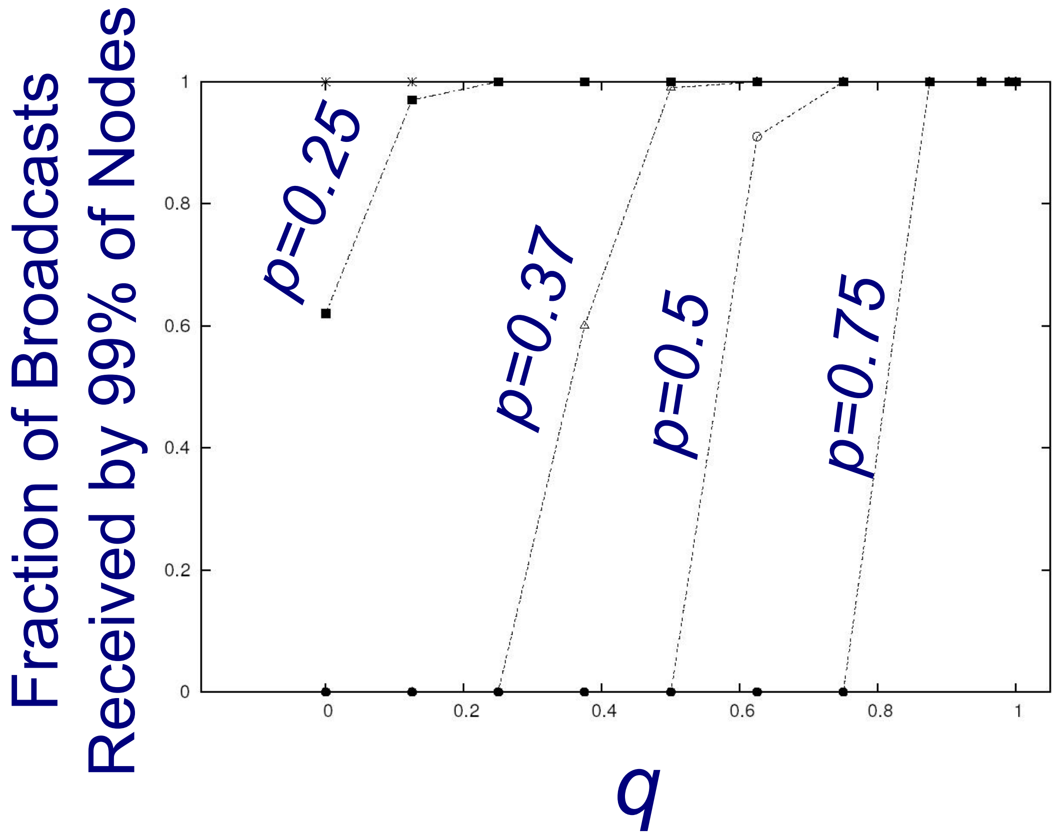


Summary of Results: Reliability

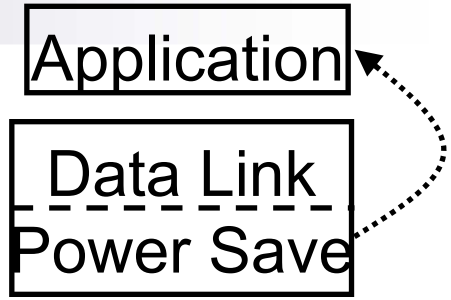
- Phase transition when:

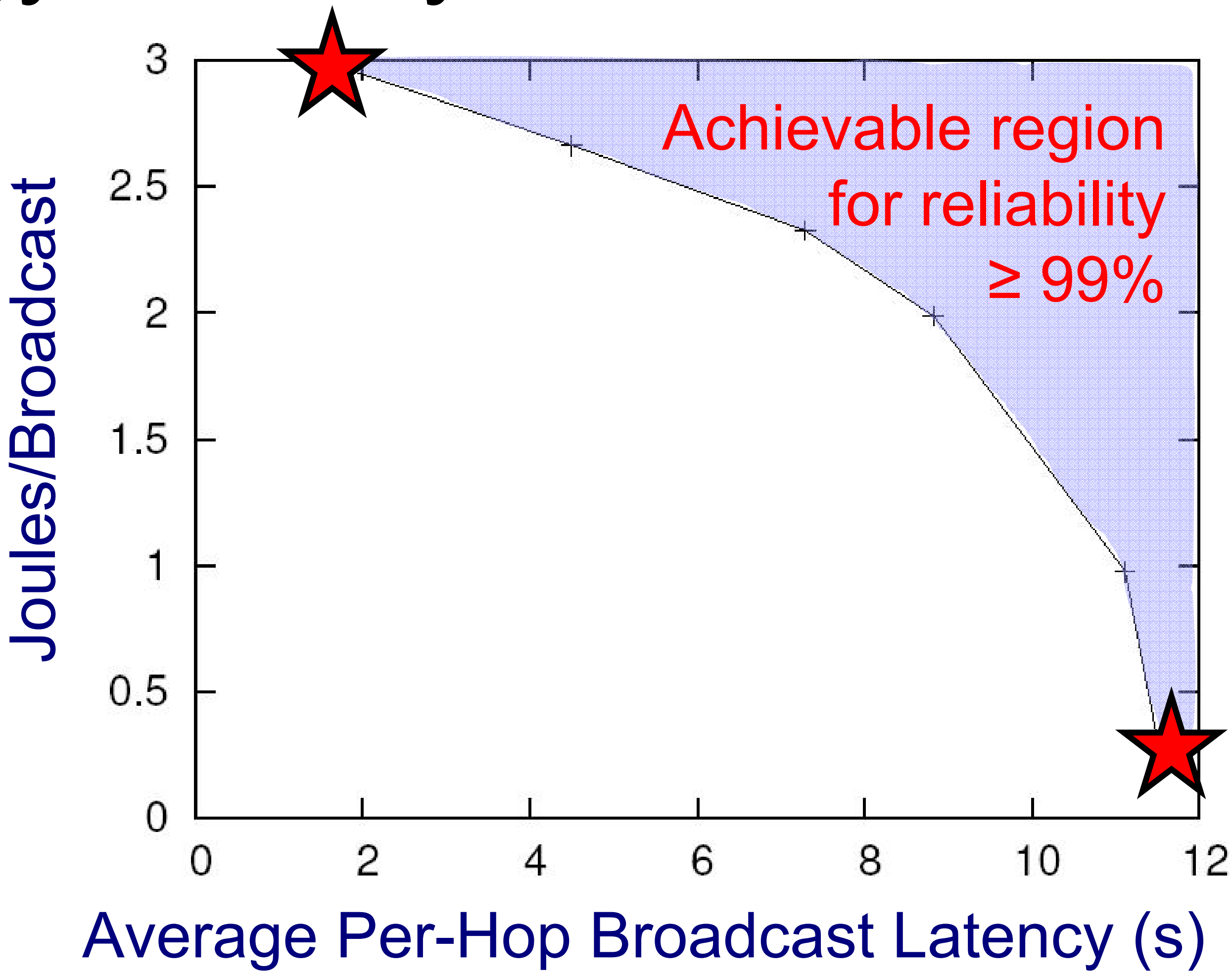
pq + *(1-p)* ≈ 0.8-0.85

- Larger than bond percolation threshold
  - Boundary effects
  - Different metric
- Still shows phase transition

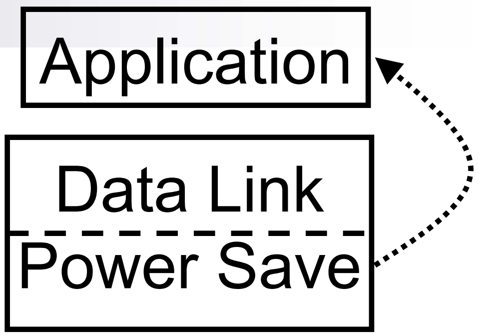


Summary of Results: Energy-Latency Tradeoff





Summary

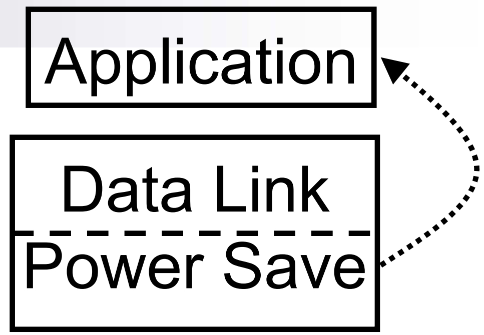


- Shown the effects of energy-saving protocols on latency and reliability of applications that disseminate data via multihop broadcast
- Designed protocol that allows wide range of tradeoffs for such applications
- *Future Work*
  - Study impact of PBBF on route discovery
  - Consider per-broadcast PBBF where parameters are set by the source for each individual broadcast
- ***Acknowledgements**: Joint work done with Cigdem Sengul and Indranil Gupta*

Talk Outline

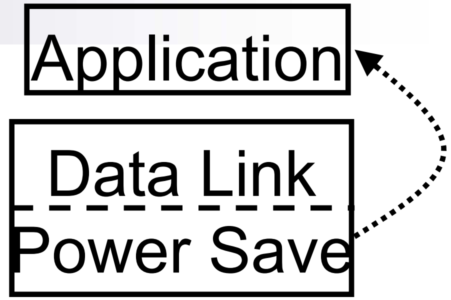
- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Neighborhood Data Sharing
- Future Work

Neighborhood Data Sharing Applications

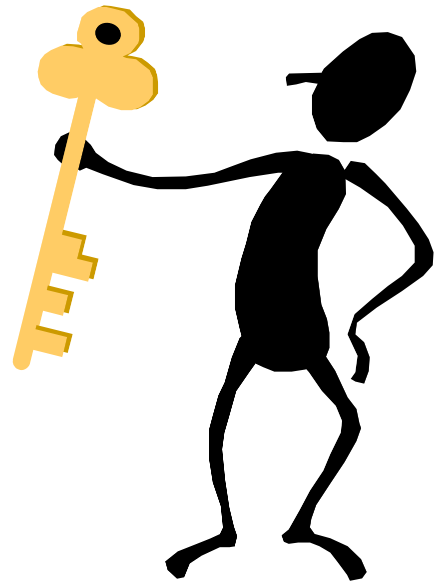


- Sharing data in a node's local neighborhood is a common method by which applications make decisions
- Example Applications
  - Proactive route updates
  - Cluster formation
  - Choosing keys for communication with neighbors
- What cross-layer effects arise in such an application as a result of power save?
  - "Quality" of a decision is application dependent
  - We focus on "quality of security" for a key distribution application

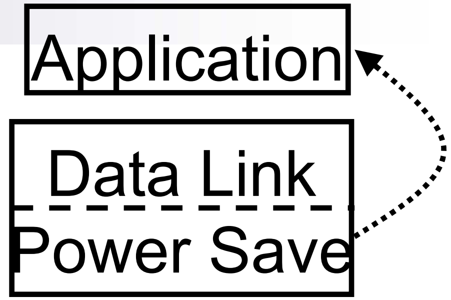
Sensor Network Security



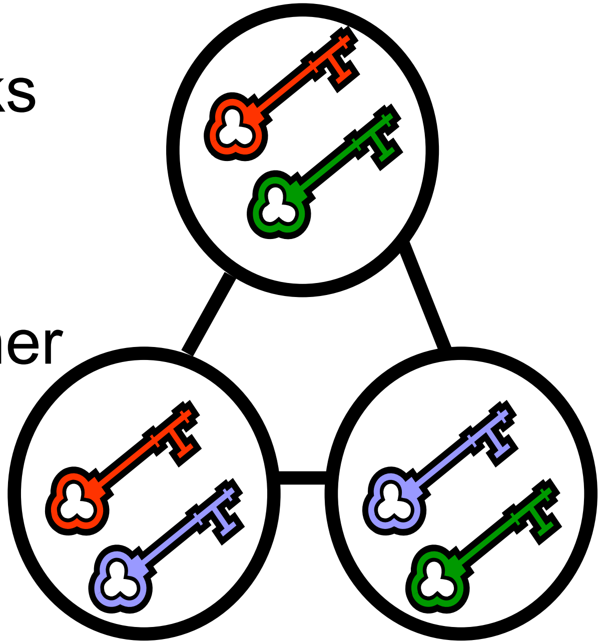
- Key distribution is an important application for sensors
  - Eavesdropping relatively easy
  - Deployment may be in hostile territory
- Challenges
  - Resource constraints
    - Use symmetric keys
    - Use little memory for keying material
  - Scalability
  - Uncontrolled topology



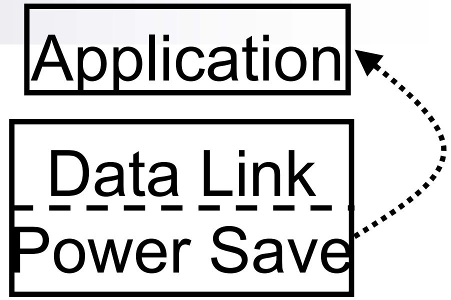
Sensor Network Key Distribution Applications



- All nodes share one key
  - Minimal memory usage
  - If one node is compromised, all links are compromised
- Separate key for each node pair
  - If one node is compromised, no other links are compromised
  - Each node must store *N* keys
- Goal: sensors share a secret pairwise key with each *one-hop neighbor* instead of *every* sensor

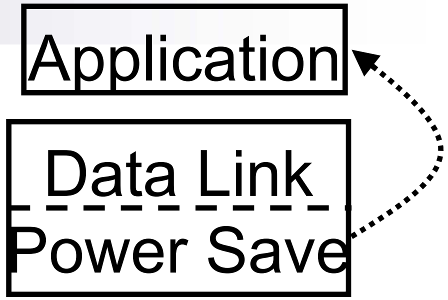


Related Work: Sensor Key Distribution



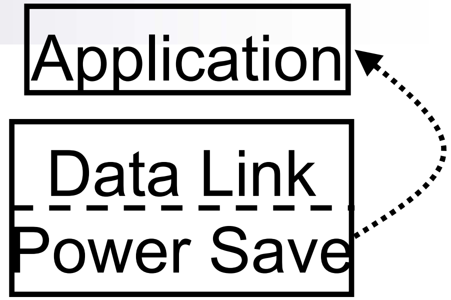
- Key Predistribution [Eschenauer02CCS] [Chan03S&P]
  - Each sensor preloaded with a random subset of keys from a global key pool
  - Sensors with shared keys can communicate
  - Relatively low connectivity and each compromised sensor exposes more of global key pool to the adversary
- [Anderson04ICNP]
  - Each neighbor pair does a plaintext handshake over the broadcast channel to establish a shared key
  - Assumes attackers are very sparse (e.g., < 3% of nodes)
  - Weaker than our protocol; does not use channel diversity

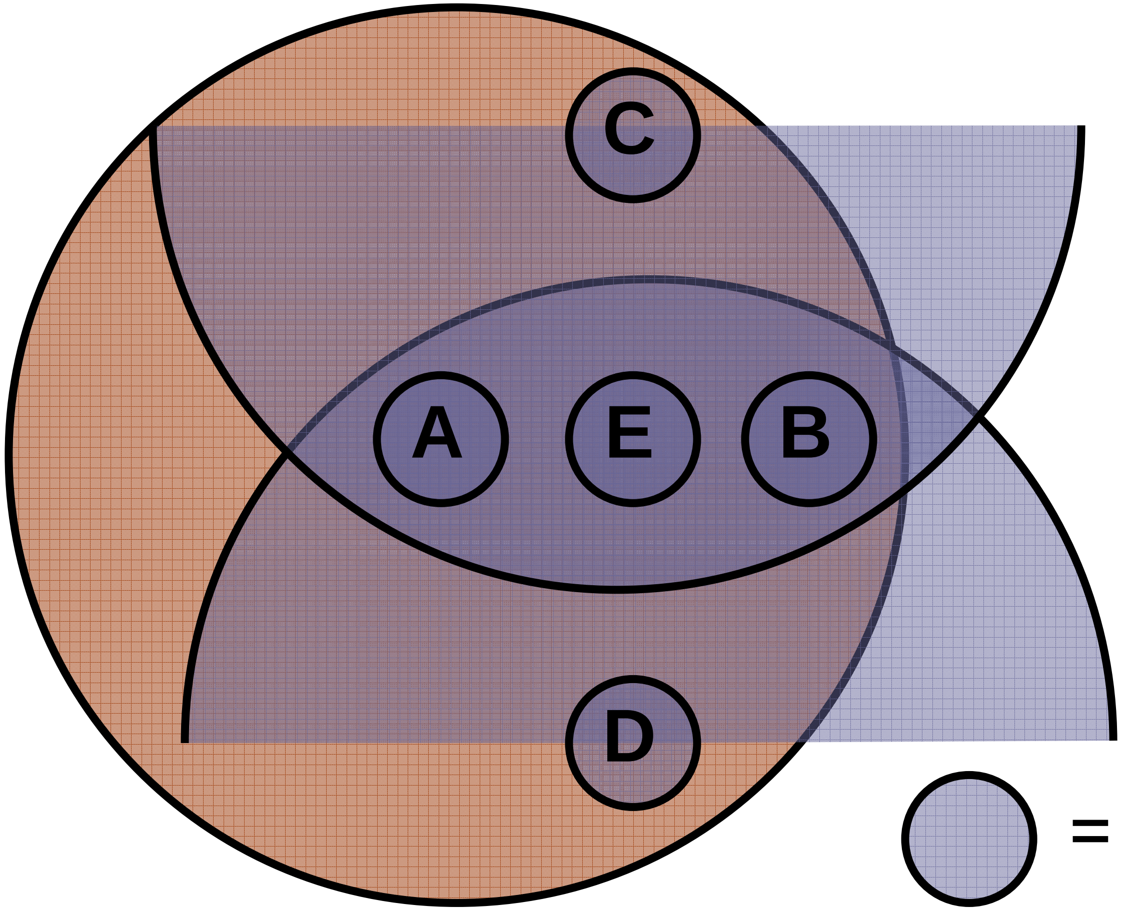
Our Work

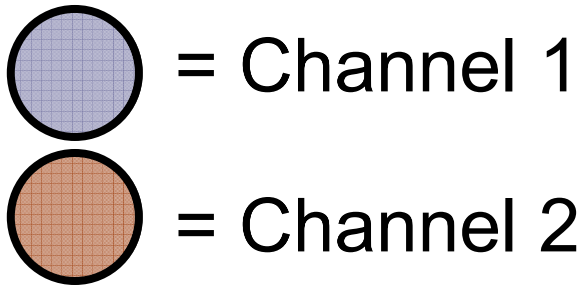


- Present novel protocol where each node stores one key per neighbor and each key is secret (w.h.p.) after short initialization
- First to propose leveraging channel diversity for sensor network key distribution
- Characterize the energy-security tradeoffs possible with our application

Basic Idea of Application







Nodes that know all of A and B's keys:

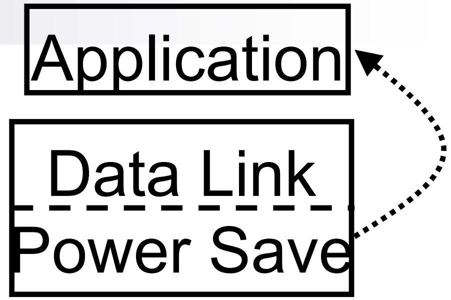
~~C, D, E~~

~~C, E~~

~~E~~

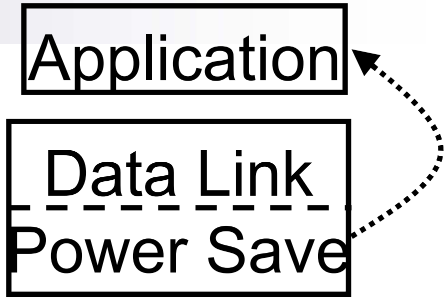
Ø

Phase 1: Predeployment



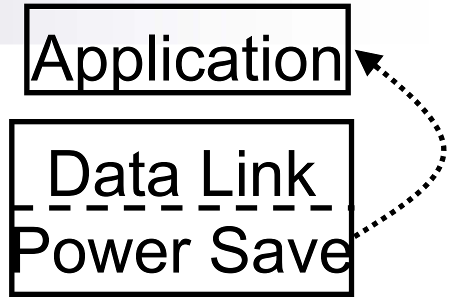
- Each sensor is given *α* keys by a trusted source
  - Keys are unique to sensor and *not* part of global pool
  - *α* presents a tradeoff between initialization overhead and security
- Given *N* sensors, the trusted source also loads O(lg *N*) Merkle tree hashes needed to authenticate a sensor's keys
  - Discussed in detail in the proposal document

Phase 2: Initialization



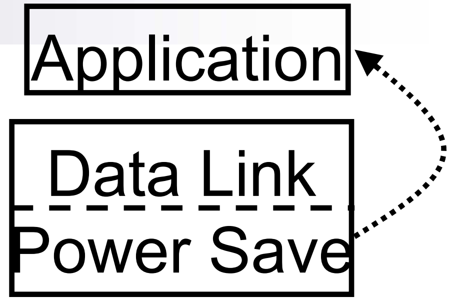
- Each sensor follows two unique non-deterministic schedules:
  - When to switch channels (chosen uniformly at random among *c* channels)
  - When to broadcast each of its *α* keys
- Thus, each of a sensor's *α* keys is overheard by *1/c* neighbors on average and a different subset of neighbors overhears each key
- Sensors store their *α* keys along with every overheard key

Phase 3: Key Discovery



- Goal: Discover a subset of stored keys known to each neighbor
- All sensors switch to common channel and broadcast Bloom filter with η of their stored keys
  - Bloom filters described in detail in proposal document
- Sensors keep track of the subset of keys they believe they share with each neighbor
  - May be wrong due to Bloom filter false positives

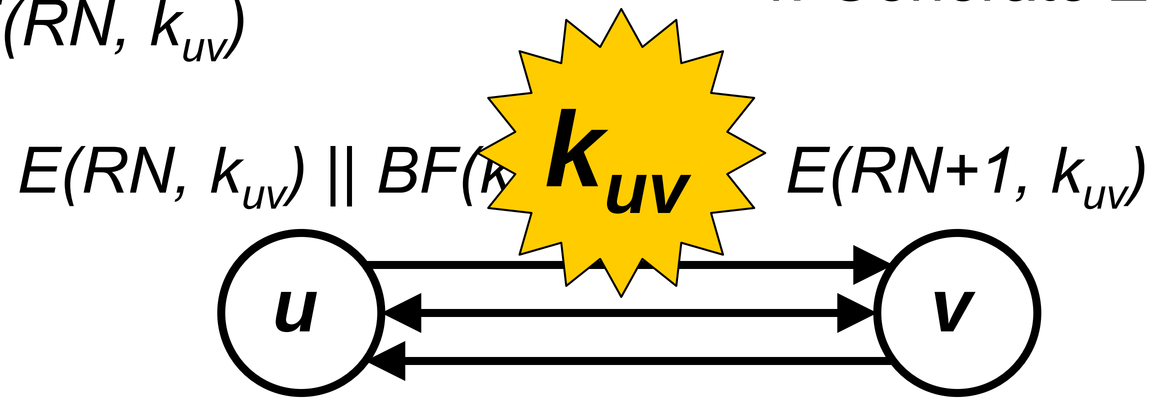
Phase 4:
Key Establishment



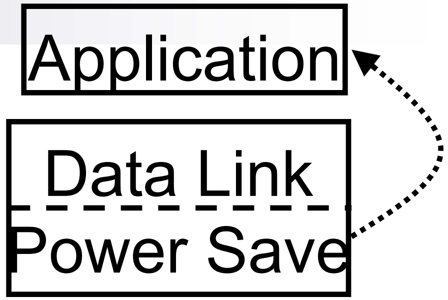
$\eta = 3$: k_1, k_2, k_3

1. Generate link key:
$k_{uv} = \text{hash}(k_1 \,||\, k_2 \,||\, k_3)$
2. Generate Bloom filter for k_{uv}: $BF(k_{uv})$
3. Encrypt random nonce (RN) with k_{uv}: $E(RN, k_{uv})$

1. Find keys in $BF(k_{uv})$
2. Use keys from Step 1 to generate k_{uv}
3. Decrypt $E(RN, k_{uv})$
4. Generate $E(RN+1, k_{uv})$

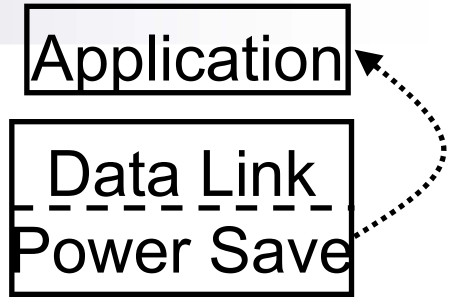


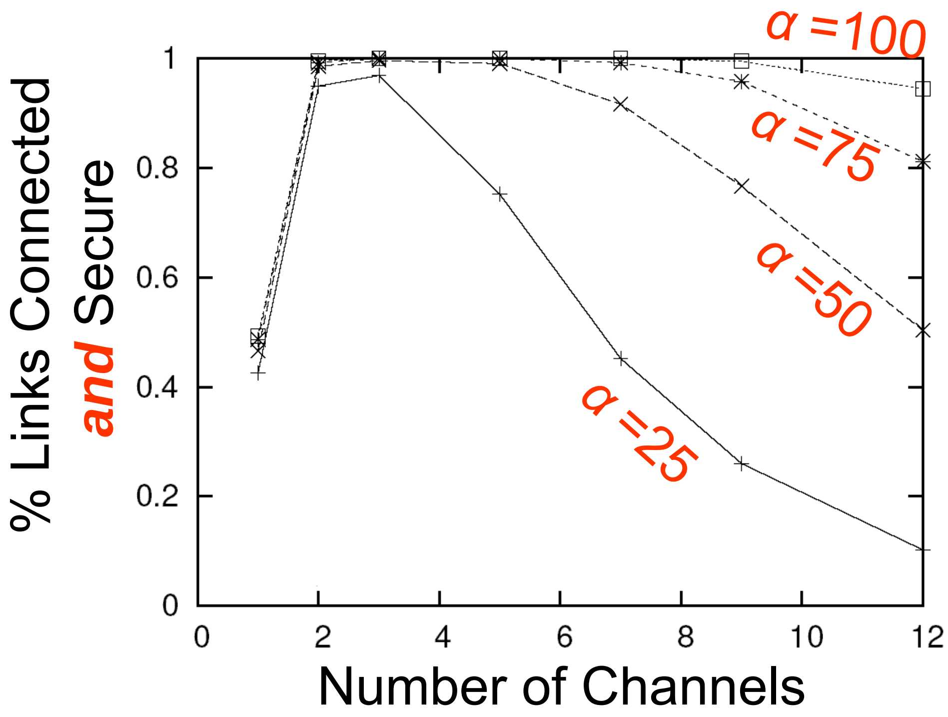
Phase 4: Key Establishment

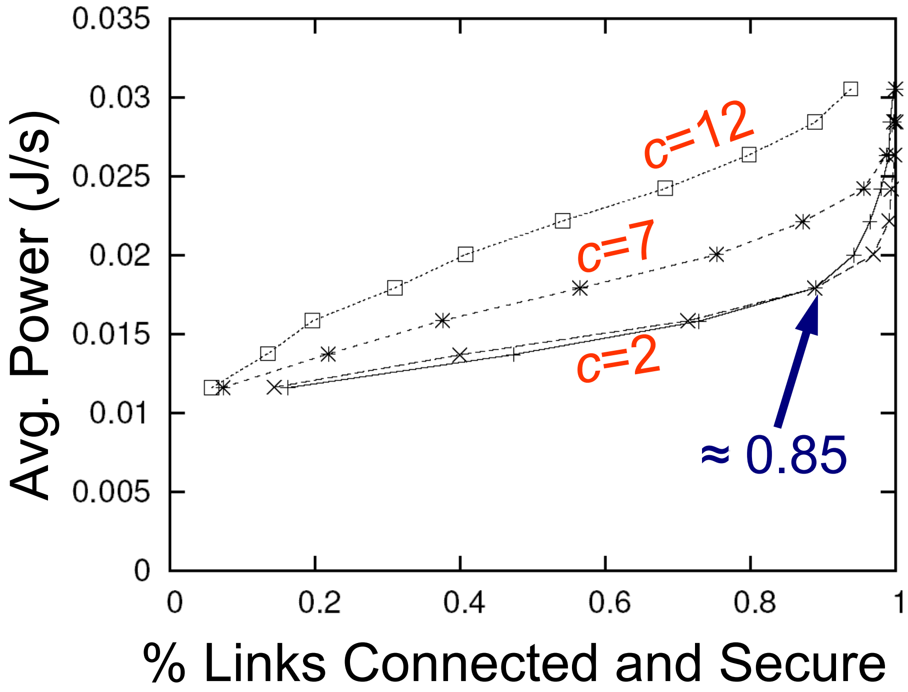


- Goal: Establish one link key with each neighbor based on subset of shared keys
- For *u* to form a link key with *v*, it first creates key k_{uv}, formed from the η keys that *u* believes it shares with *v*
- *u* then sends a Link Request to *v* with a random nonce encrypted by k_{uv} and a Bloom filter of the keys that make up k_{uv}
- *v* replies with a Link Reply if it is able to correctly decrypt the random nonce and k_{uv} is established as the link key

Summary of Results



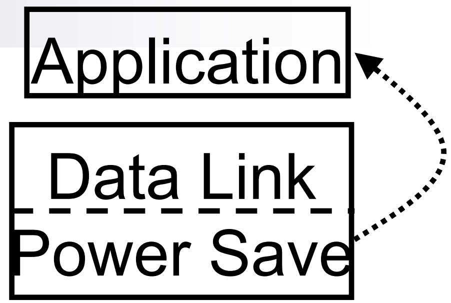




One extra channel greatly improves security

For *c*=2, small energy increase greatly improves security when % less than about 0.85

Summary



- Designed key distribution application for sensor networks that is resilient to compromise and has relatively low memory requirements
- Unlike other protocols, we leverage channel diversity as part of the protocol design
- Characterized the cross-layer security-energy tradeoffs that arise when sensors use power save with the application

Talk Outline

- Background on Energy Efficiency
- Link Layer/Physical Layer Design
- Link Layer/Routing Layer Design
- Cross-Layer Effects on Multihop Broadcast
- Cross-Layer Effects on Key Distribution
- Future Work

Future Thesis Work: Routing

- Simulate and evaluate proposed latency-aware routing protocol
- Simulate and evaluate proposed network-lifetime aware routing protocol
- Integrate these protocols with proposed physical layer CS protocol

Future Thesis Work: Multihop Broadcast

- Study impact of PBBF on route discovery
  - Effect on quality of routes since nodes receive less route requests
- Study per-broadcast PBBF
  - Parameters are set by the source of each individual broadcast rather than using the same values for every network broadcast

Thank You!!!

https://netfiles.uiuc.edu/mjmille2/www/research.htm

mjmille2@crhc.uiuc.edu

Analysis in Our Work

- Developed equations to model energy consumption of all 4 OOB protocols in the physical layer CS chapter
- Analysis for key distribution protocol to determine the probability that a pairwise key is secret
- Co-authors for PBBF used percolation theory to model energy consumption, latency, and reliability

Properties of Preamble Sampling

- No synchronization necessary
  - We require synchronization
- Larger preambles increase chance of collisions
  - We restrict CS signals to a time when data is not being transmitted
  - In our technique, interference is tolerable between CS signals
- Broadcasts require preamble size be as long as a BI → Exacerbates broadcast storm
  - We do not require extra overhead for broadcast
- Only one sender can transmit to a receiver per BI
  - We allow multiple senders for a receiver per BI

Is time synchronization a problem?

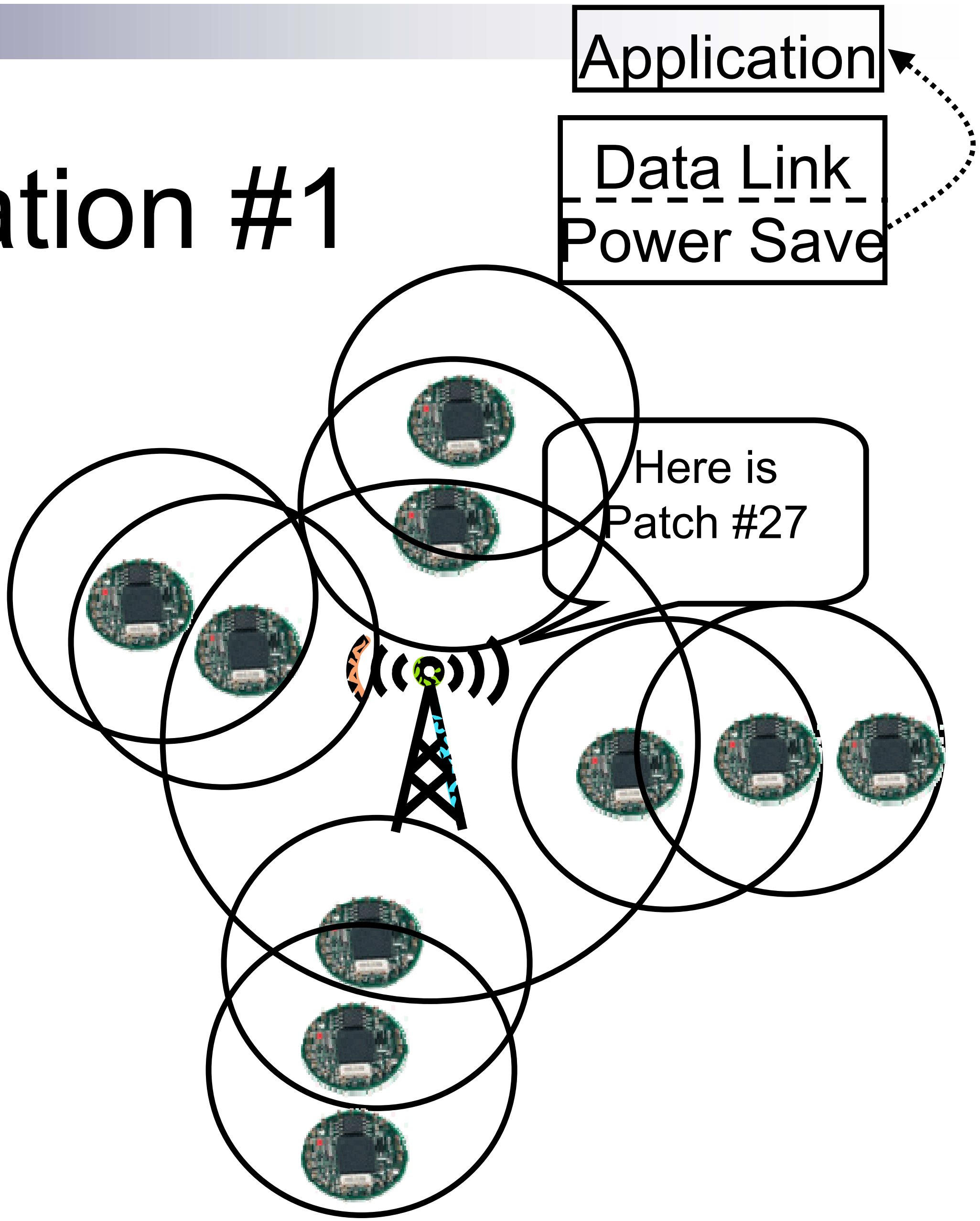
- Motes have been observed to drift 1 ms every 13 minutes [Stankovic01Darpa]
- The Flooding Time Synchronization Protocol [Maróti04SenSys] has achieved synchronization on the order of one microsecond
- Synchronization overhead can be piggybacked on other broadcasts (e.g., routing updates)
- GPS may be feasible for outdoor environments
- Chip scale atomic clocks being developed that will use 10-30 mW of power [NIST04]

Transition Costs Depend on Hardware [Polastre05IPSN/SPOTS]

Mote Radio Model	**Wakeup Time (ms)**	**TX/RX/ Sleep (mW)**	**Bitrate (kbps)**
TR1000 (*1998-2001*)	0.020	36/12/ 0.003	40 ASK
CC1000 (*2002-2004*)	2	42/29/ 0.003	38.4 FSK
CC2420 (*2004-now*)	0.580	35/38/ 0.003	250 O-QPSK

Sensor Application #1

- Code Update Application
  - E.g., Trickle [Levis et al., NDSI 2004]
- Updates Generated Once Every Few Weeks
  - Reducing energy consumption is important
  - Latency is not a major concern



Sensor Application #2

- Short-Term Event Detection
  - E.g., Directed Diffusion [Intanagonwiwat et al., MobiCom 2000]
- Intruder Alert for Temporary, Overnight Camp
  - Latency is critical
  - With adequate power supplies, energy usage is not a concern