Sensor Medium Access Control (S-MAC)

Copied from “*Robin Dunn*” smac.ppt

Outline

- **Motivation**
- **Protocol design**
  - **Periodic Listen & Sleep**
  - **Adaptive Listening**
  - **Overhearing Avoidance**
  - **Message Passing**
- **Protocol Implementations**
- **Experimentation**
- **Summary**
- **References**

Motivation

In a WSN:

“energy conservation and self-configuration are primary goals...”[1]

Motivation

Need to reduce energy waste from:

- **Collision**
- **Overhearing**
- **Control overhead**
- **Idle Listening**

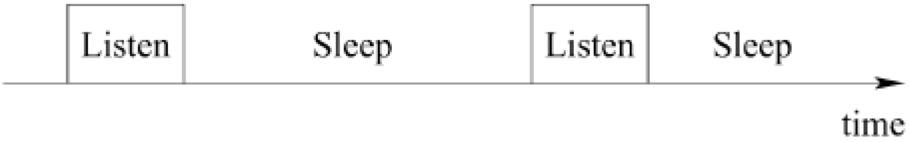
Need to have a robust topology to:

- **Account for lost nodes**
- **New joining nodes**

Periodic Listen and Sleep(1)

Choosing & Maintaining Schedules

- **Nodes choose their own schedules or follow others**

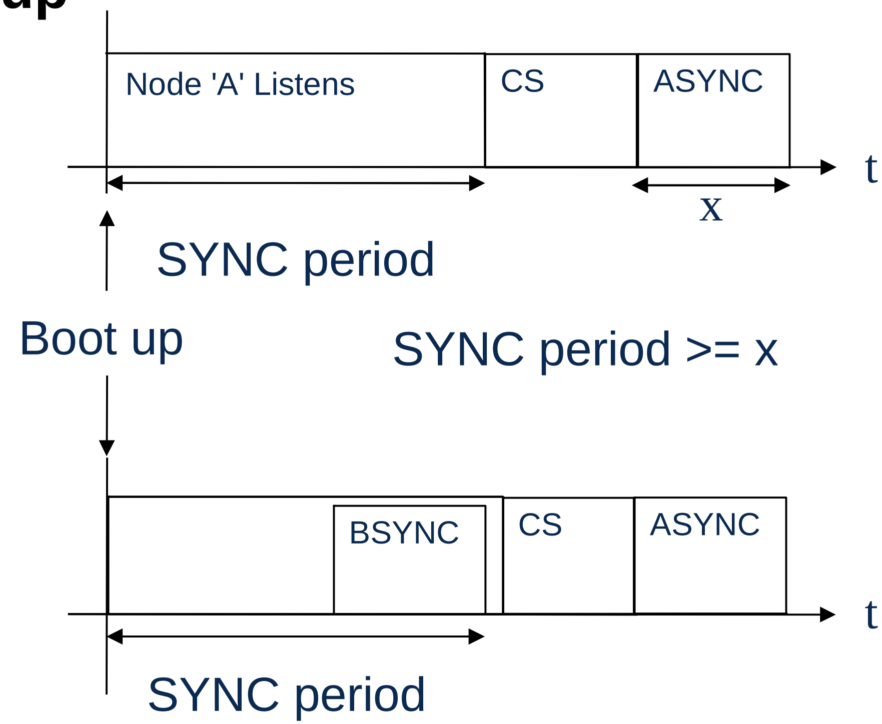


[2]

- **Neighbouring nodes synchronise – virtual clusters**
  - Reduces latency
  - SYNC packet to communicate schedules
    - Address of sender
    - Time to next sleep
  - Duration of sleep not transmitted – user defined at compile time
  - Node exchange schedules by broadcasting it to all its neighbours (ensures all neighbors can talk)
  - Schedules of neighbours stored in schedule table

Choosing & Maintaining schedules

- **Boot up**



Receives no SYNC so chooses and follows own schedule

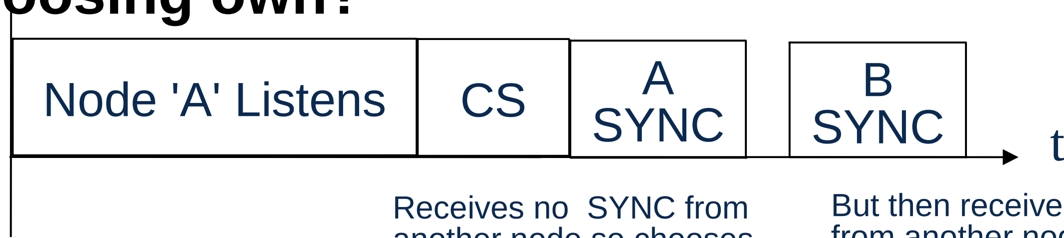
Receives SYNC from node B so follows B's schedule

When a node has chosen or adopted a schedule it broadcasts its schedule.

Periodic Listen and Sleep(2)

Choosing & Maintaining Schedules

– **What if a node receives a different schedule after choosing own?**

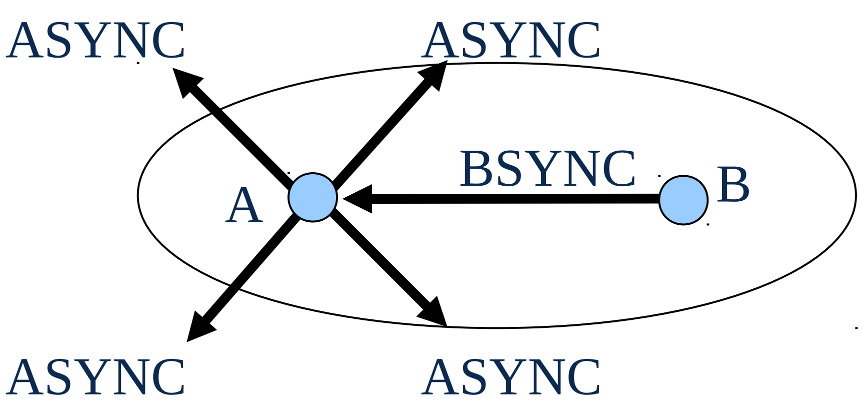


Receives no SYNC from another node so chooses and follows own schedule

But then receives SYNC from another node B.

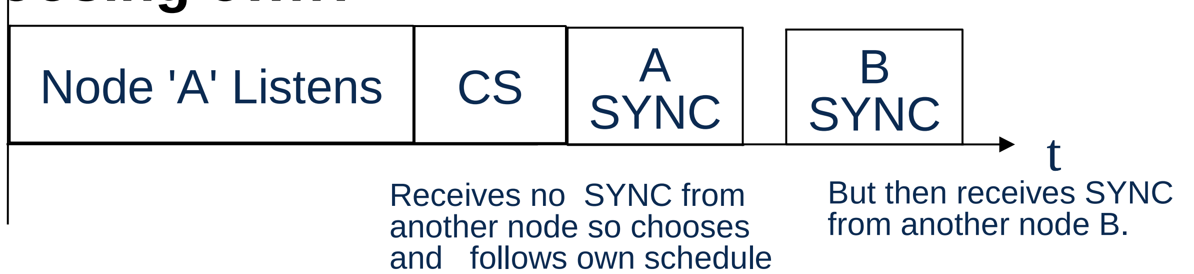
If A has no other neighbours:

1. A listens...hears no SYNC...
2. A Chooses own schedule...
3. Broadcasts SYNC...
4. B joins...
5. B broadcasts own SYNC...
6. A adopts B's schedule.



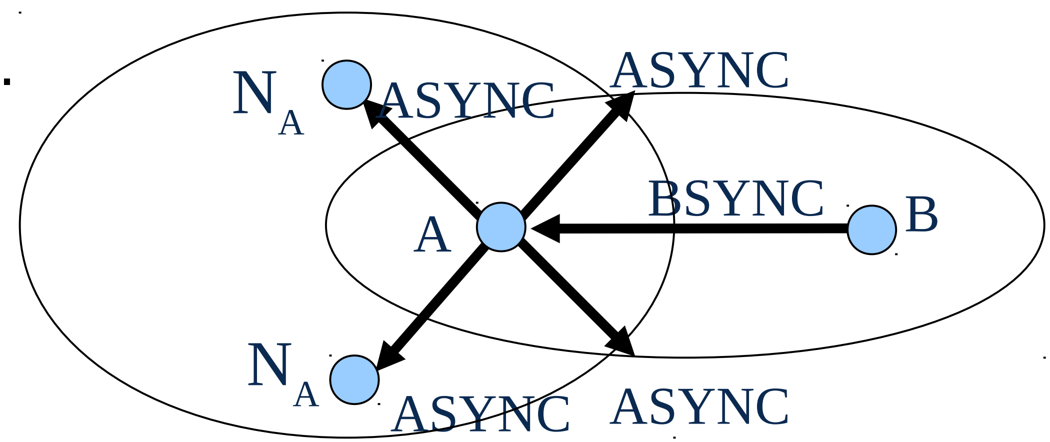
Choosing & Maintaining Schedules

– **What if a node receives a different schedule after choosing own?**



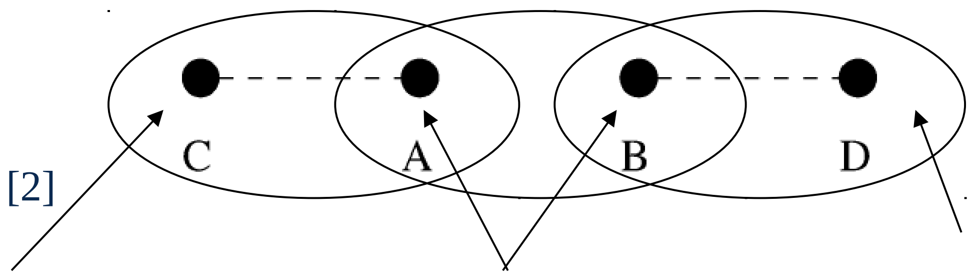
If A has at least one neighbour:

1. A listens...hears no SYNC...
2. A Chooses own schedule...
3. Broadcasts SYNC...
4. A following schedule with neighbours...
5. B joins...
6. B broadcasts own SYNC...
7. A adopts both schedules.



Periodic Listen and Sleep(3)

Choosing & Maintaining Schedules



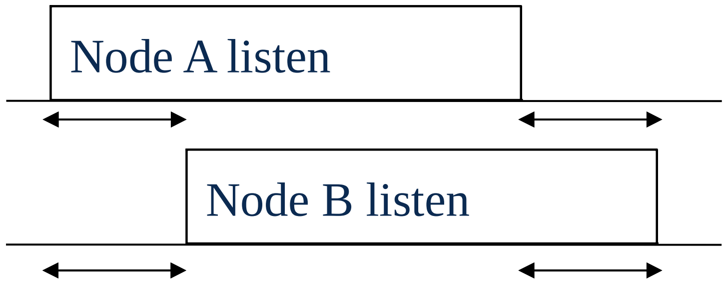
[2]

- Form virtual clusters but communicate with peers
- No Cluster head.
- Border nodes have less time to sleep.
- Periodic neighbour discovery
  - nodes periodically listen for the whole SYNC period

Periodic Listen and Sleep(4)

Maintaining Synchronization

- Clock drift → synchronisation errors
- Avoid these errors using
  - Relative Timestamps – time to next sleep
  - Listen times longer than clock drift rate

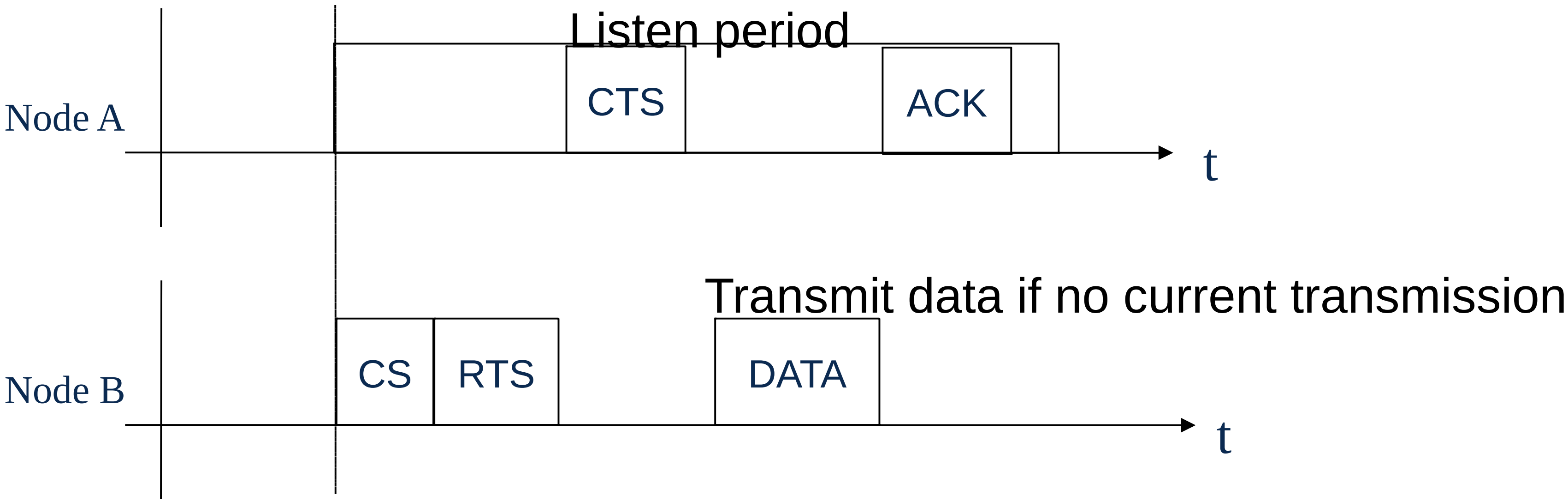


  - Periodic synchronisation updates.
    - SYNC packet (same as on Boot Up).
      - Address of sender
      - Time of senders next sleep

Periodic Listen and Sleep(5)

Timing for sending and Receiving data

- **Contention**
  - Node B wants to send data to node A



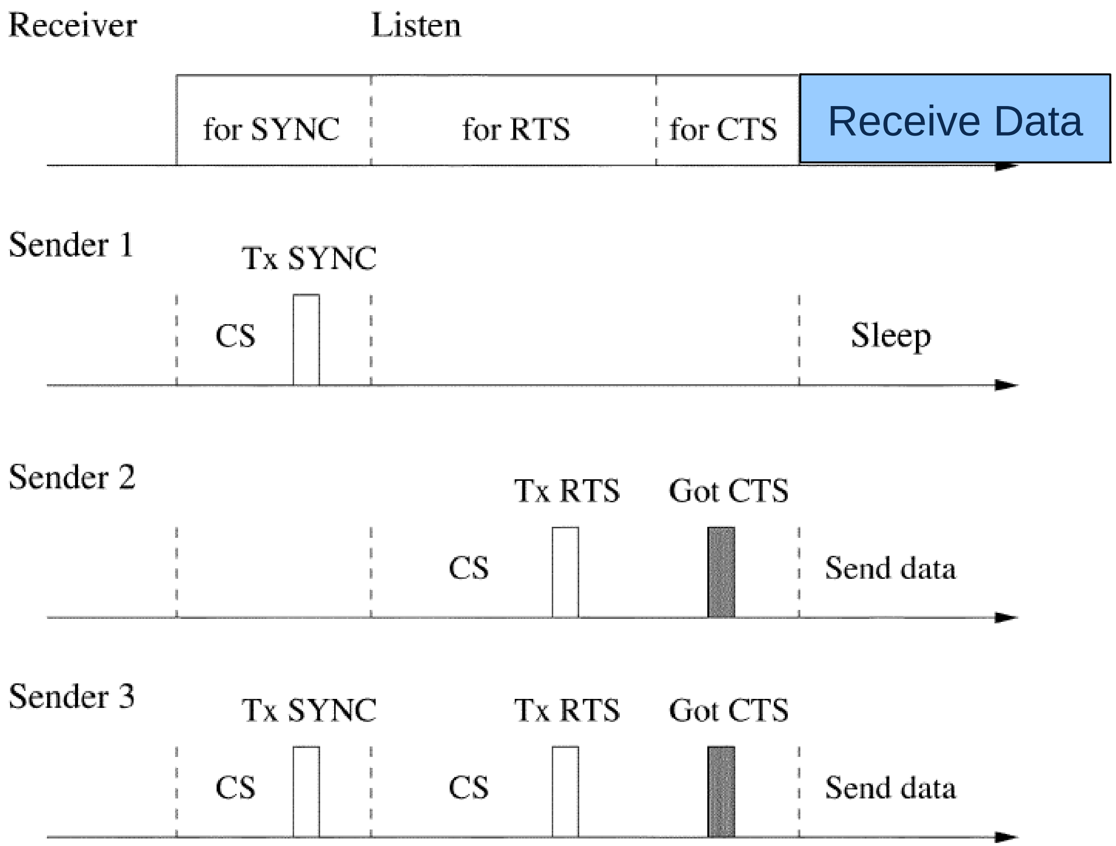
CS = Carrier Sense RTS = Request to send CTS= Clear to send

Carrier Sense is for random duration

Periodic Listen and Sleep(6)

Timing for sending and Receiving data

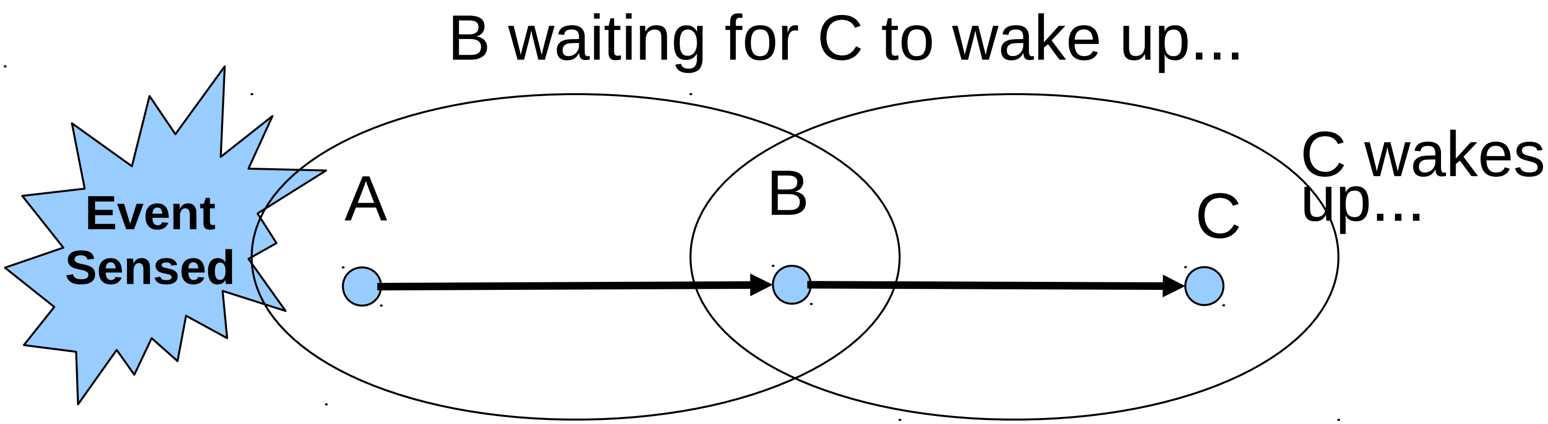
- Division of the listen interval
  - To ensure both SYNC and Data packets are received



[2]

Adaptive Listening - Problem

- **Delay caused by strict adherence to sleep schedules**

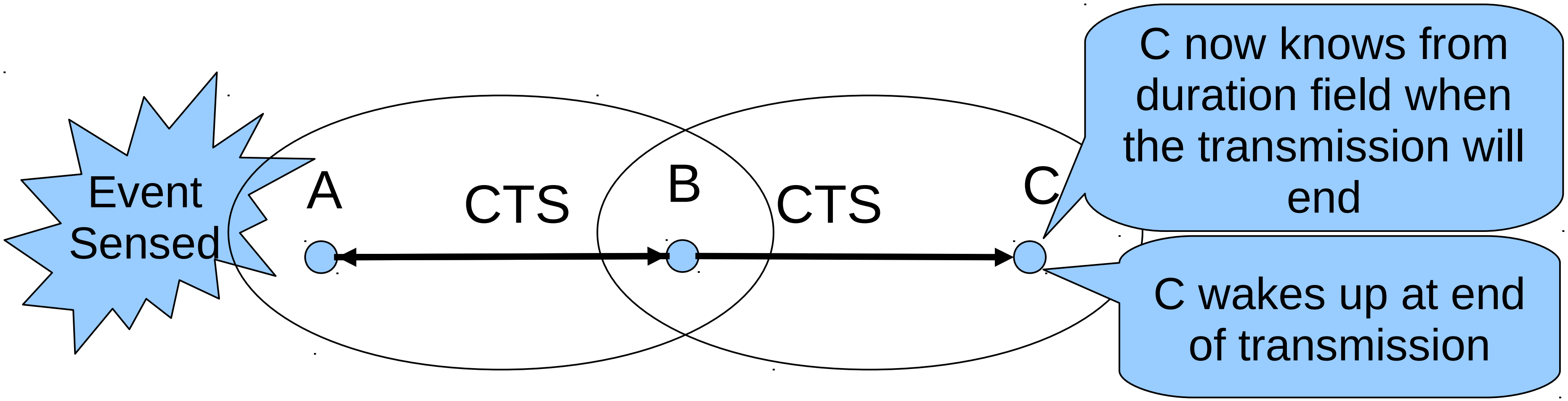


B wants to pass the message on to C. It knows when C's next listen time is but it doesn't want to wait until then. It wants to send the message immediately!

Adaptive Listening - Solution

Adaptive Listen mechanism to reduce delay

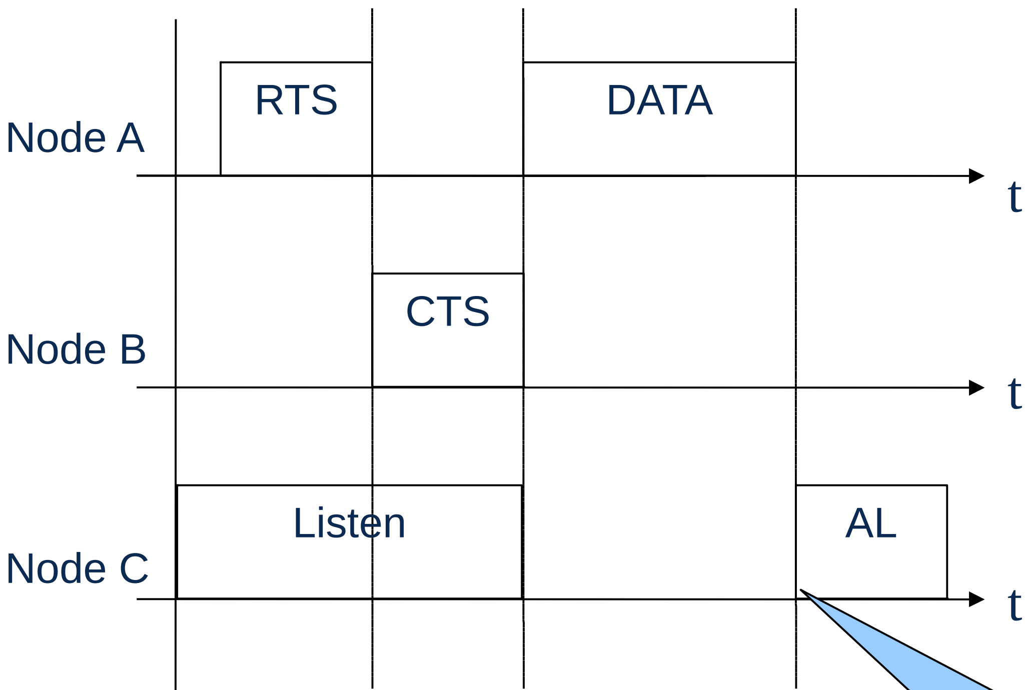
- Duration field included in control packets (RTS/CTS/ACK)

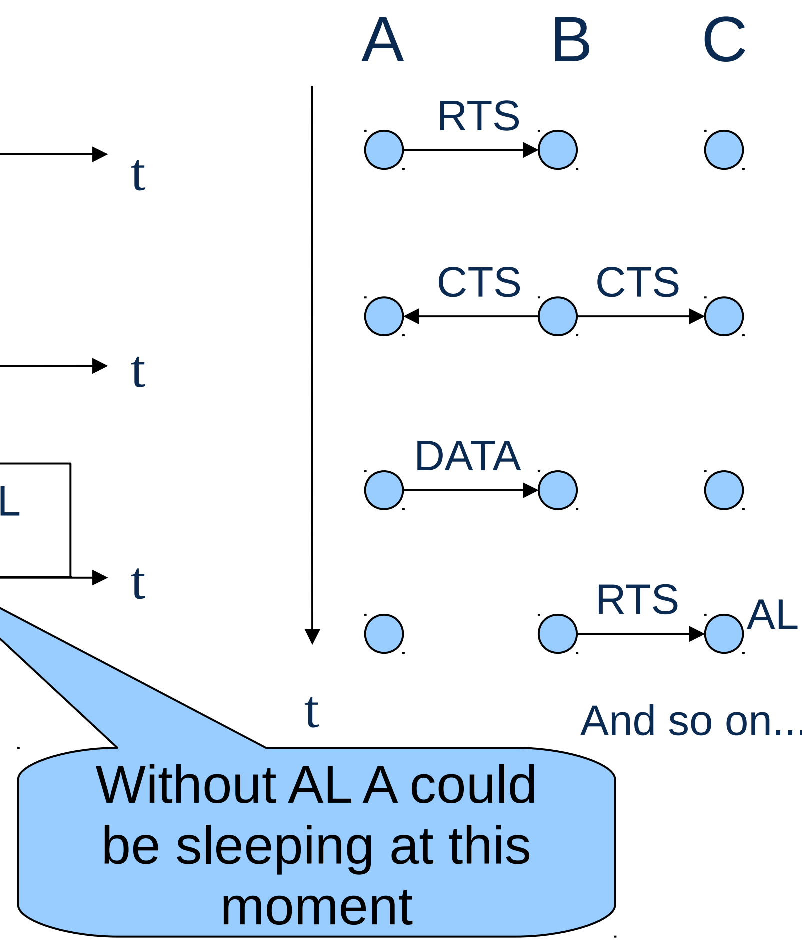


- Network Allocation vector (NAV) – count down to end of transmission
- As soon as B has received all the data from A, C will wake up immediately to receive the data from B
- Receives RTS from B and so on...
- If C missed RTS/CTS it can get duration from ACKs

Adaptive Listening

- **Adaptive Listen mechanism to reduce delay - summary**

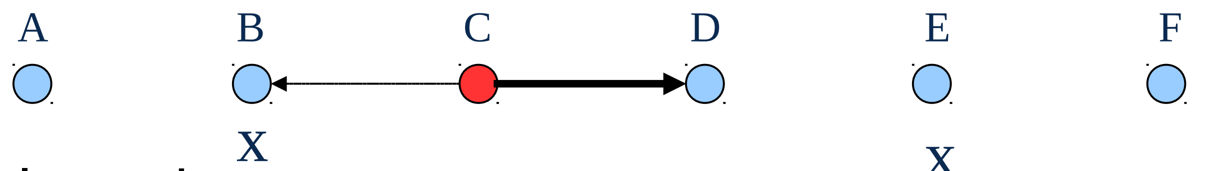




RTS = Request to send CTS= Clear to send
AL = Adaptive Listen

Overhearing Avoidance

- **Problem:** A node listens to data transmissions even though it knows it is not involved.
- **Solution:** Nodes sleep after RTS, CTS or ACK
- Which nodes sleep?



[2]

- B sleeps because:
  - A→B collides with C→B (RTS,DATA)
  - B→A collides with D→C (ACK)

- E sleeps because:
  - E→D collides with C→D (RTS,/DATA)

- All immediate neighbours of transmitting node sleep
- Nodes know when to wake up by using NAV

Message Passing(1)

Efficient transmission of long messages.

- Other Approaches:

One long packet.

Corruption of part means resending all.

Many small packets with RTS,CTS,DATA,ACK for each.

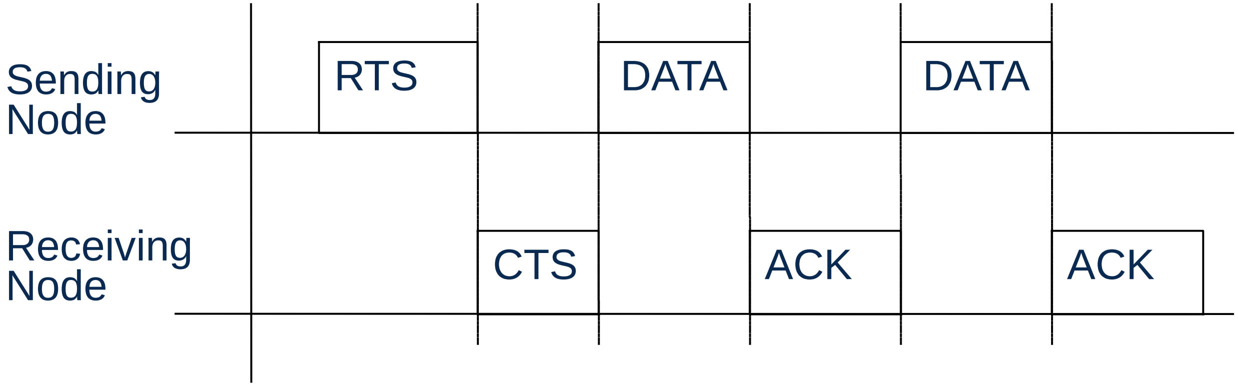
Large control overhead and delay though contention for each.

Message Passing(2)

Message Passing Approach:

- Fragmented message transmitted as burst

1RTS. 1CTS. Multiple ACKs.

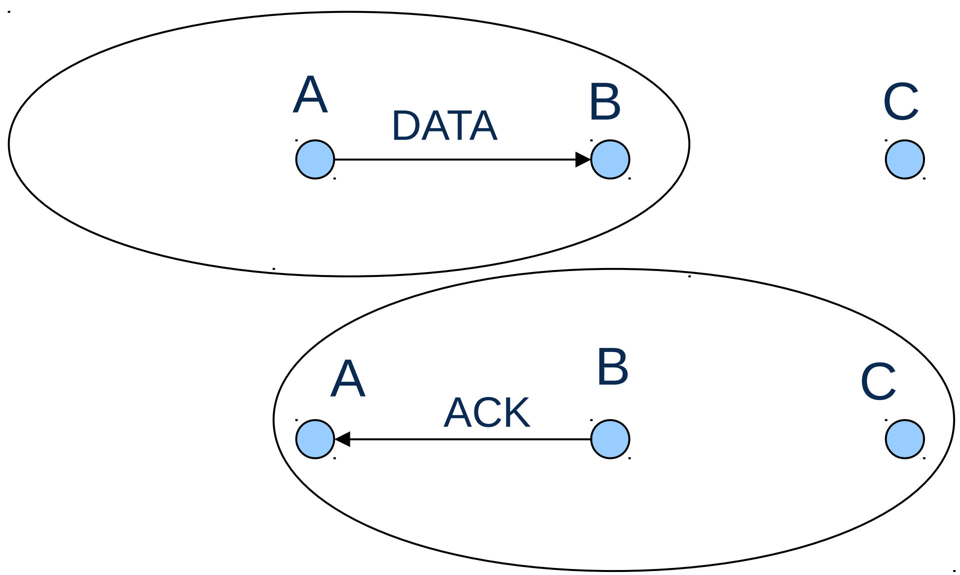


- Extension of transmission time for resending of corrupted packets
- Limit on number of extensions to account for dead receiver.
- Multiple ACKs required for AL and overhearing avoidance

Message Passing (3)

Advantages

- Smaller delay when packets are corrupted
  - i.e. Can resend just corrupted packets
- Smaller control overhead
- Favours message fairness over node fairness
- Sending multiple ACKs prevents hidden terminal problem:



- C cant here A's transmission
- C thinks B is free

- C hears ACK
- Realises B is busy
- Sets NAV

Implementation

- 2 Implementations carried out:
  - 1st Rene Motes (No Adaptive Listening). 2 hop network. [1]
  - 2nd Mica Motes (Adaptive Listening Option) 10 hop network.[2]
  - Both using TinyOS

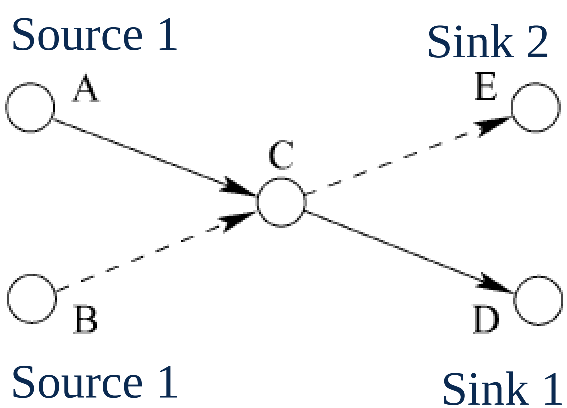
- User can configure options when compiling:
  - Duty Cycle selection (1%-99%)
  - Fully Active Mode (periodic sleep disabled)
  - Disable adaptive listen

- Current Implementations coordinate radio sleeping.
- Further work required to put other hardware to sleep (e.g. CPU).
  - Further energy savings

Experimentation(1)

1st Implementation.

- Rene Motes
- 2 hop network.



[2]

Biggest energy savings when low traffic

Mean energy consumption on radios in each source node

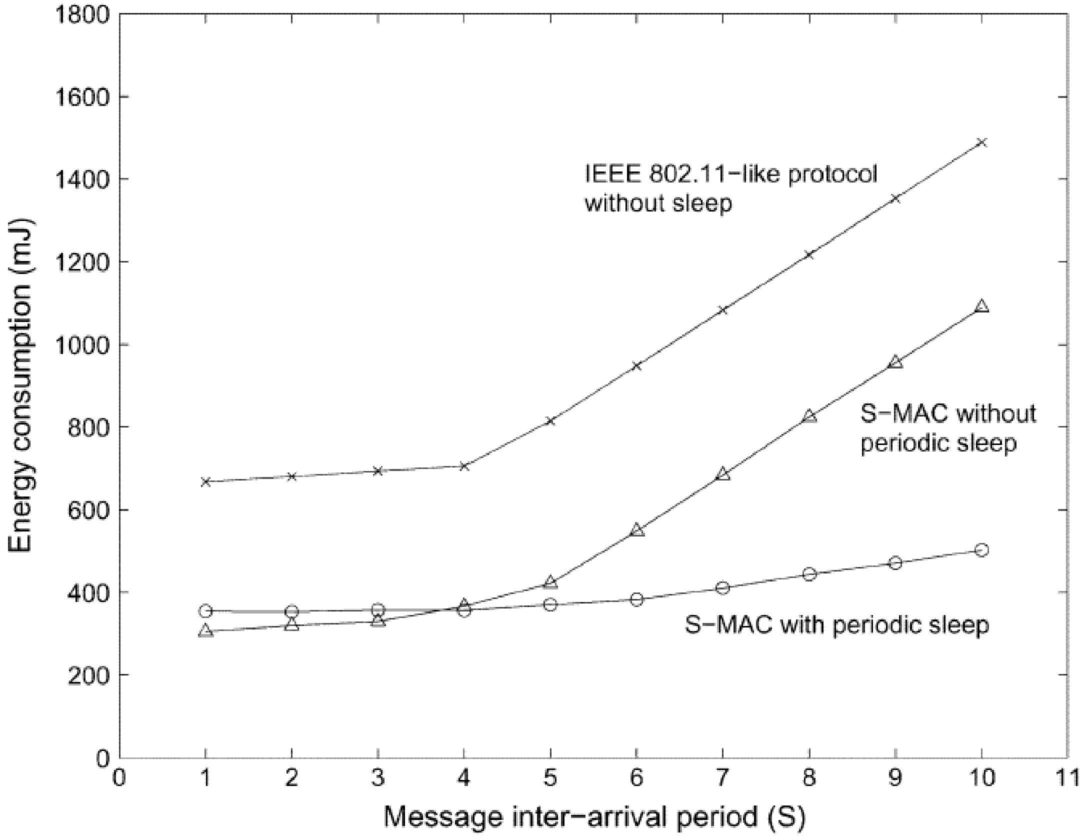


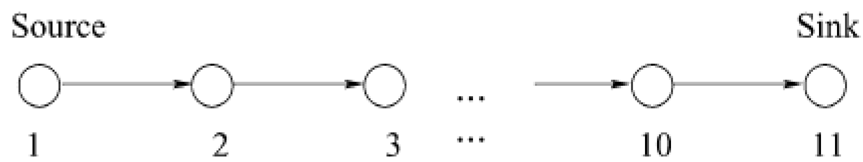
Fig. 8. Mean energy consumption on radios in each source node.

[2]

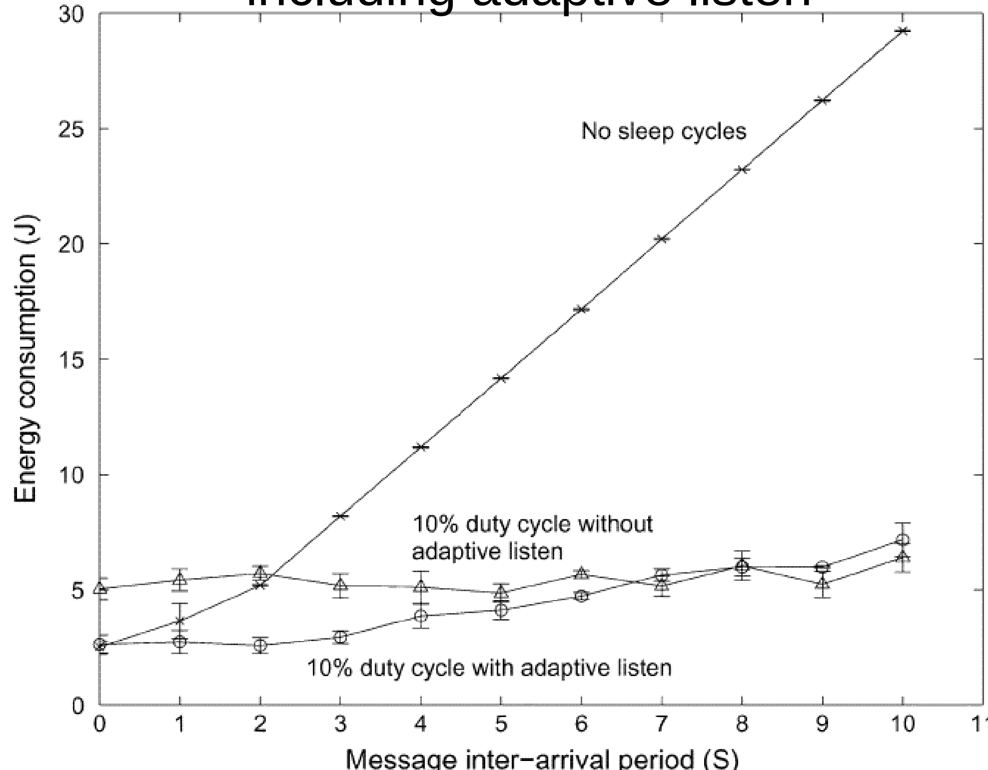
Experimentation(2)

2nd Implementation

- Mica Motes
- 10 hop network



Energy consumption results including adaptive listen [2]



[2]

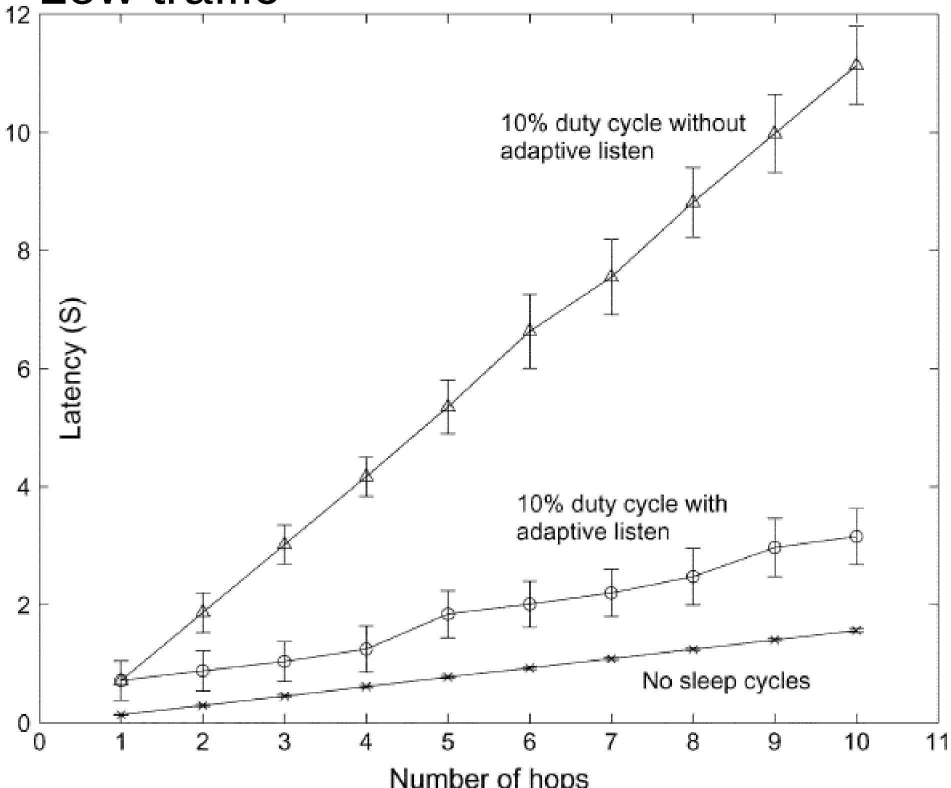
Fig. 10. Aggregate energy consumption on radios in the entire ten-hop network using three S-MAC modes.

Again biggest energy savings during low traffic

Experimentation (3)

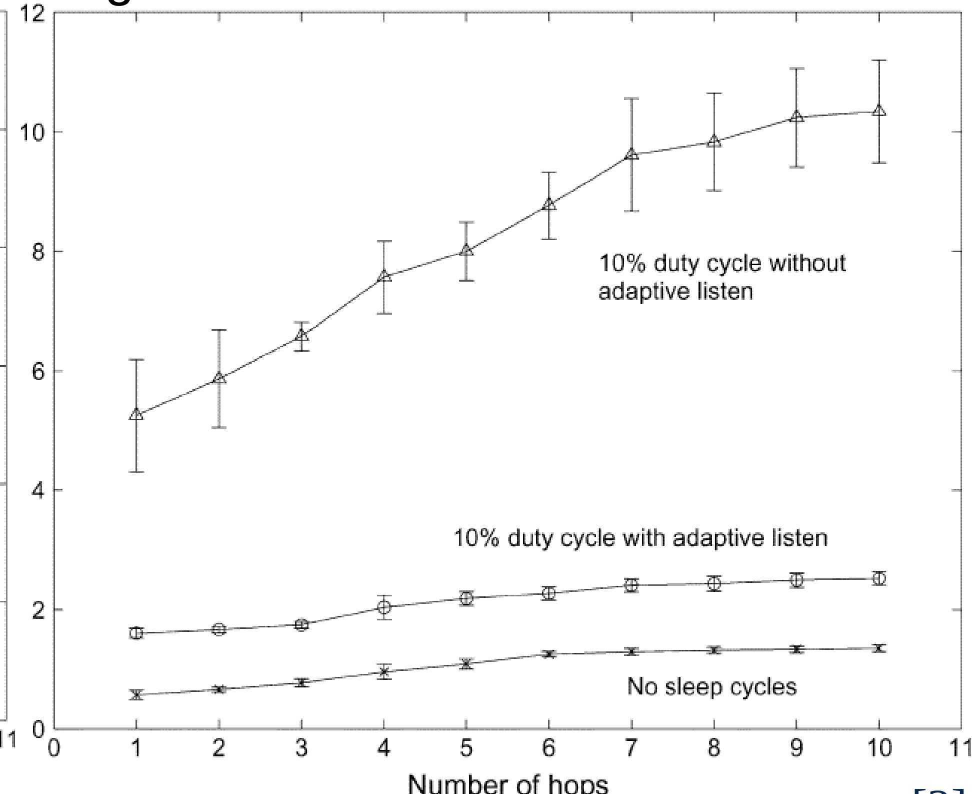
Latency - measured from the time a message is generated on the source node

Mean message latency on each hop - Low traffic



[2]

Mean message latency on each hop - High traffic



[2]

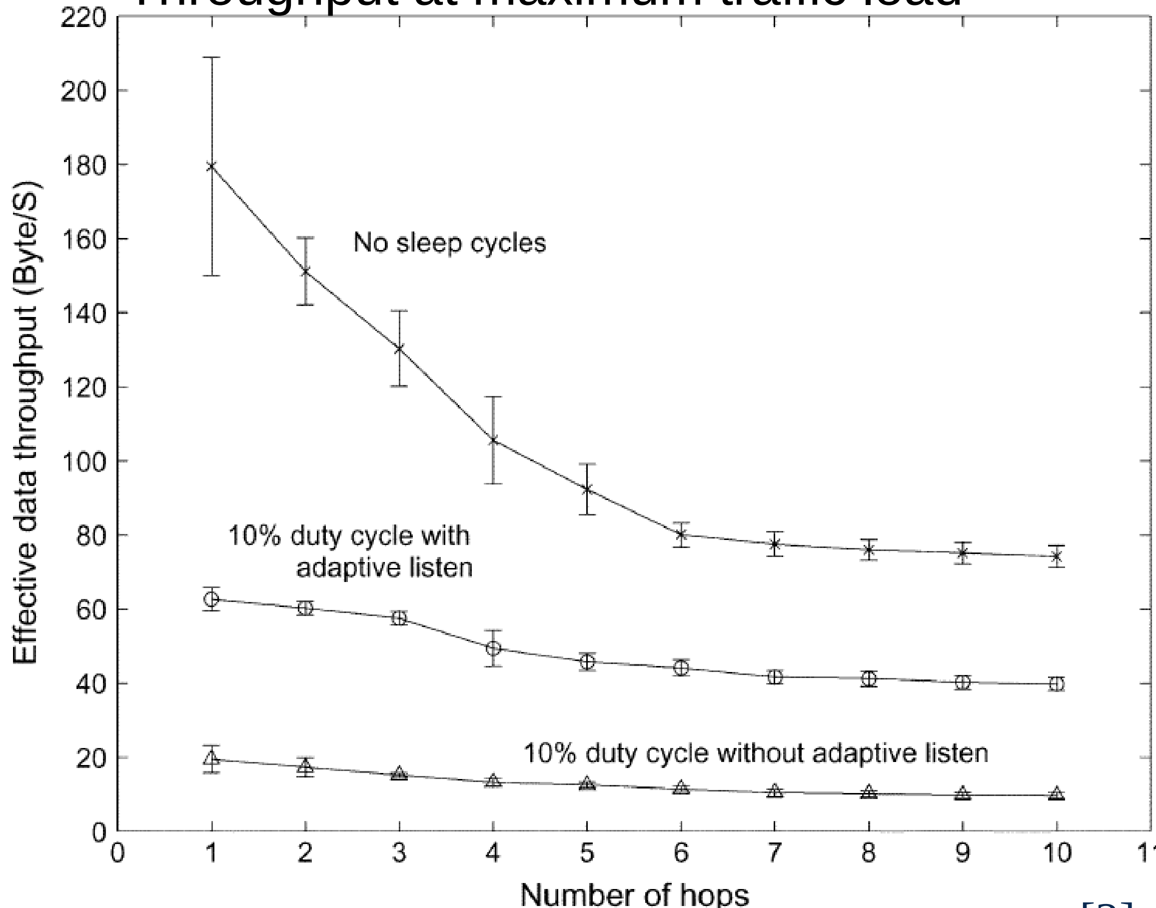
AL – Reduces latency

Experimentation(4)

Throughput

– only data packets counted, not control packets.

Throughput at maximum traffic load



AL improves throughput

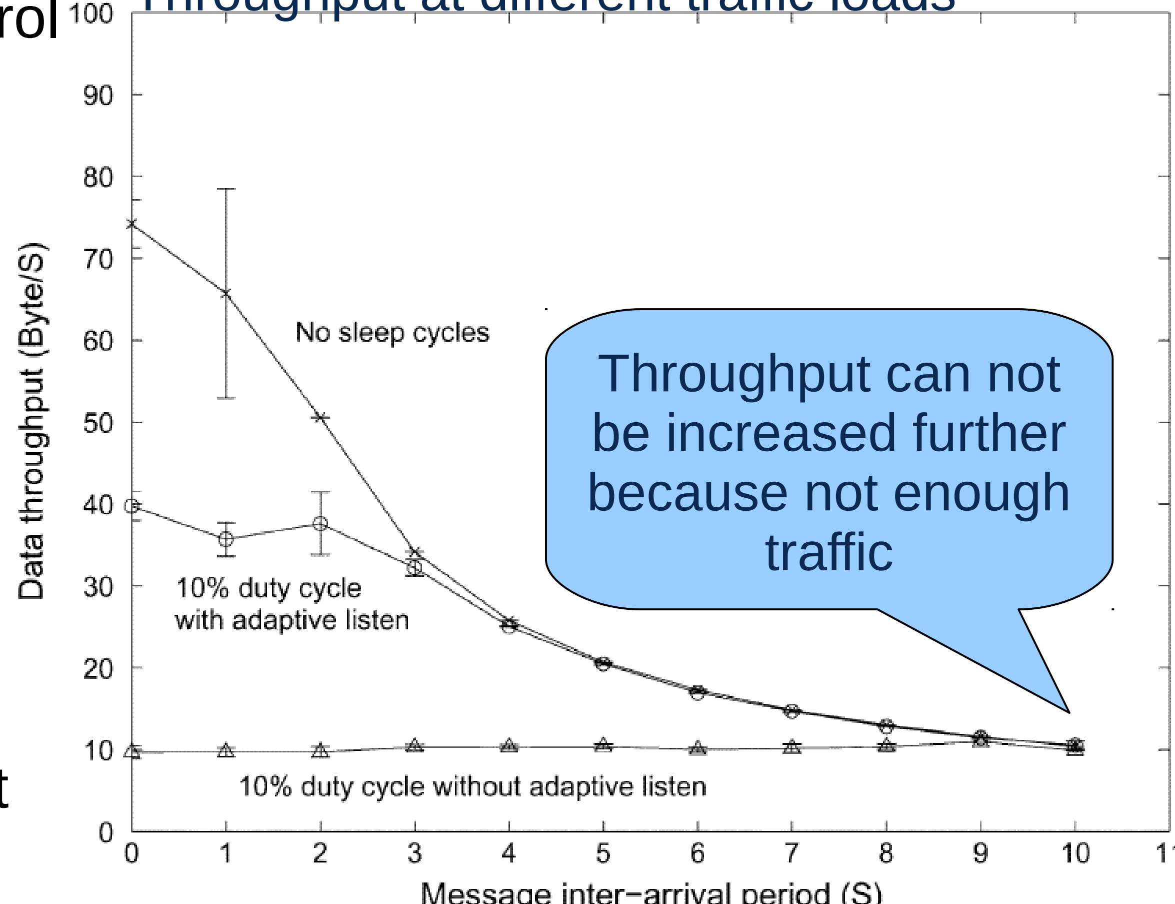
[2]

Experimentation(5)

Throughput

– only data packets counted, not control packets.

Throughput at different traffic loads

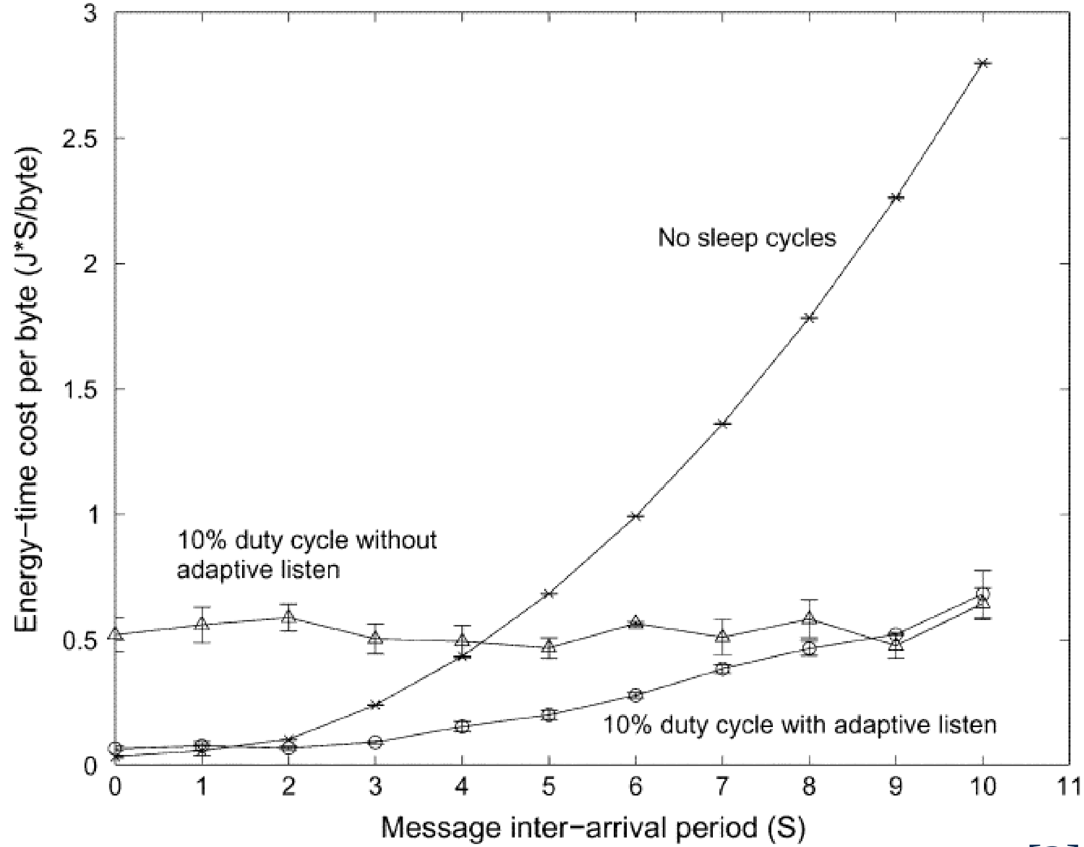


[2]

AL improves throughput

Experimentation (6)

Combined affect of energy consumption and reduced throughput



[2]

Fig. 15. Energy–time cost per byte on passing data from source to sink under different traffic load.

The benefits of adaptive listen occur at moderate to high traffic load.

Summary

- **SMAC makes energy savings by periodically putting nodes to sleep.**
- **This can result in higher latency and reduced throughput**
- **But these problems are mitigated by the use of Adaptive Listening**

Summary

- Motivation
- Protocol design
  - Periodic Listen & Sleep
  - Adaptive Listening
  - Overhearing Avoidance
  - Message Passing
- Implementations
- Experimentation
- Summary
- References

References(1)

Main references:

- [1] *An Energy-Efficient MAC Protocol for WirelessSensor Networks*. Wei Ye, John Heidemann, Deborah Estrin.
  In Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Vol.3, pp. 1567-1576, New York, NY, USA, June, 2002.
- [2] *Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks.* Wei Ye, John Heidemann and Deborah Estrin
  IEEE/ACM Transactions on Networking, Vol. 12, No. 3, pp. 493-506, June 2004.

- *SMAC Source code available to download at*
  http://www.isi.edu/ilense/software/smac/download.html

References(2)

Related Work:

- *Energy and Latency Control in Low Duty Cycle MAC Protocols. Yuan Li, Wei Ye, John Heidemann. IEEE Wireless Communications and Networking Conference (WCNC), March 2005, New Orleans, LA.*
- *Implementation of the Sensor-MAC protocol for the JiST/SWANS simulator.* Dissertation. Tippanagoudar, Veerendra. FLORIDA ATLANTIC UNIVERSITY. 2006. http://wwwlib.umi.com/dissertations/preview/1436493 *Have to pay to get full access!*
- *SMAC Implementation for ETRI SSN-based Sensor Networks. Kiran Tatapudi and Chansu Yu Department of Electrical and Computer Engineering Cleveland State University.* *http://www.csuohio.edu/ece/techReport/2006/tr03.pdf*
- *Adaptive Forwarding Error Correction Code Control: An Adaptive FEC Code Control Algorithm for Mobile Wireless Sensor Networks. JOURNAL OF COMMUNICATIONS AND NETWORKS, VOL. 7, NO. 4, DECEMBER 2005.* *http://network.dongguk.ac.kr/*