Adaptability in the B-MAC+ protocol

Marco Avvenuti and Alessio Vecchio

PERLAB - Pervasive Computing Lab
Dip. Ingegneria dell'Informazione
University of Pisa, Italy
Energy Consumption in Wireless Communication

With current technology, a large fraction of energy consumption is due to communication:

• Communication rate
  − requirement of the application

• Idle Listening
  − listening for potential transmissions

• Overhearing
  − receiving packets addressed to other nodes

90% of the overall consumption for several applications
Power-aware communication

• An effective power saving solution is to switch the radio on and off, cyclically

• **Duty cycling** can be done efficiently at the MAC layer, provided that nodes have overlapped tx and rx periods:
  
  - **Schedule-based** protocols rely on a global schedule and reserved tx/rx slots, so that senders know when receivers are on
  
  - **Contention-based** protocols are asynchronous and rely on preamble sampling, so that senders can be listened by receivers (even though they poll the channel un-frequently)

• In the following we will focus on a contention-based, low power listening protocol
The B-MAC protocol

- Receivers periodically poll the channel (check interval)
- Senders must use “wake-up” preambles to bridge the gap between two consecutive polls
- When a receiver listens to the channel:
  - if active, the radio is held on to receive the remaining part of the preamble and the data payload
  - if idle, the radio is switched off
Idle listening is reduced but:

- Recipients must receive (possible long) preambles while waiting for the payload
- Non-recipients overhear preambles
The idea: replace the flat bit pattern of the wake-up preamble with a sequence of information blocks

Senders divide the preamble into chunks. Each chunk contains:
- a counter, the size of the remaining part of the preamble
- an address, the actual recipient of the data packet

Receivers use these infos for:
- turning the radio off for the remaining part of the preamble
- early rejection of a packet
Saving power with B-MAC+

Lightweight reception

Receiver ‘a’
sleep

Sender
sleep

check
check
check

extra sleep

7 6 5 4
3 2 1 0

Receiver ‘b’
sleep

Sender
sleep

check
check
check

early rejection

7 6 5 4
3 2 1 0

DATA

sleep
check
check

DATA

extra sleep

early rejection
B-MAC vs B-MAC+

Node 1 (transmitter)

Node 2 (receiver)

Node 3 (neighbour)

Wake-up preamble

Data tx

Transceiver is turned on

Channel check

Overhearing

Block tx

Data tx

Saved energy

Block rx

Data rx

Channel check

Block rx
Setting the check interval is NOT an easy task

- Long check intervals reduce idle listening (receivers are turned on less frequently), but:
  - senders spend more energy to send longer preambles
  - under certain traffic conditions, the additional energy spent by senders can overwhelm the energy saved by receivers

- Nodes are subject to different traffic loads depending on position and time:
  - in data gathering applications the network is usually a convergecast tree; nodes closer to the sink are subject to heavier traffic
  - when an event is detected the network is characterized by an increased activity
Need for adaptation

Lifetime of a WSN under low/high traffic load versus the check interval

- The lifetime has a maximum (Ta, Tb)
- The optimal check interval depends on traffic load
- The lower the traffic, the larger the optimal check interval (less energy spent by senders)
Need for adaptation

Comparing network lifetime:

- ideal adaptation to traffic rate
- non-adaptive solution (optimized for 1 packet/minute traffic rate)

To maximize network longevity:

- Up to the programmer
- Ideal adaptation: require a-priori knowledge of traffic
- Real adaptation: nodes observe traffic and change their check interval
Adapting the check interval to the traffic rate

A simple idea:

• when a node observes an increasing amount of incoming traffic, the node shortens its check interval:
  - nodes that transmit to it can save energy, as they can use shorter wake-up preambles
• to work, senders must know the current length of each recipient‘s check interval
Adapting the check interval to the traffic rate

- The check interval can only assume values from a pre-fixed set of low power listening modes: LPL0, LPL1, ..., LPLn (LPL0 corresponds to the shortest check interval and LPLn to the longest)

- Each node stores the arrival time of the last \( k \) packets and uses these values to compute an approximation of the traffic rate

- If the estimated traffic rate corresponds to an LPL lower than the current one, the node moves to the new mode (shorter check interval) and set a timeout \( T_t \)

- When \( T_t \) expires, the node moves to the higher LPL
Adapting the preamble to recipient's check interval

• Nodes maintain a *neighbours table* that, for every known neighbour, contains the estimated LPL mode and Tt

• Passive dissemination:
  - neighbours are informed about the current LPL of a node only when that node communicates
  - when a node sends a packet, it embeds its current LPL and Tt within the preamble (no overhead)
  - take advantage of packet overhearing for updating the table

• When a node wants to send a packet, it seeks the neighbour table for the check interval of the recipient node:
  - if any, the sender builds the packet with a properly dimensioned preamble
• Initially all nodes are in the highest mode (LPL8) and all the neighbour tables are empty.
An example (cont'd)

- Node1 sends a packet to Node3 and disseminates its status information (LPL8); neighbours (Node3 and Node2) update their tables.
An example (cont'd)

- On receiving a packet, Node3 switch to LPL7 (Tt=30s); then sends a packet to Node4; neighbours (Node1, 2 an 4) update their tables.

<table>
<thead>
<tr>
<th>Neighbours table</th>
<th>Node 1 LPL7</th>
<th>Node 2 LPL7</th>
<th>Node 3 LPL7</th>
<th>Node 4 LPL7</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>mode</td>
<td>time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>LPL7</td>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>LPL8</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>LPL7</td>
<td>30</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Neighbours table</th>
<th>id</th>
<th>mode</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LPL7</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Neighbours table</th>
<th>id</th>
<th>mode</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>LPL7</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>
An example (cont'd)

- After 4s, Node2 sends a packet to Node3, using a shorter preamble.

Node2 can save power as it knows Node3’s status.
Reverting to higher LPLs (longer check intervals) is triggered by the timeout, both in the recipient and in the sender side

• Sender side:
  - use of too short preambles is prevented by the timeout
  - use of too long preambles only affects performance (power, delay), but still guarantees packet delivery

• Recipient side:
  - if the node never communicates, it might happen that senders believe a check interval longer than the current one, but this only affects performance
Experimental evaluation

• The adaptive B-MAC+ has been implemented as a component of the TinyOS/TmoteSky platform

• The testbed was an array of three nodes, N1-N2-N3:
  - 1 packet every 10s and, periodically, a packet burst were sent from N1 to N3 and return
  - nodes introduced a 100ms delay to simulate data processing
  - N1 injected packets and measured the RTT of packets
Increasing traffic:

- on packet arrival, the receiver switches to LPL modes with shorter check intervals
- the sender adapts the length of preambles
Decreasing traffic:

- triggered by the timeout, the receiver switches to LPL modes with longer check intervals (packet arrival may cause a temporary transition)
- accordingly, the transmitter extends the length of preambles
Experimental evaluation (cont'd)

- Packet burst occurred between packets 25 and 45
- When the transmission rate increases, the packet delay decreases

Adaptation of the duty cycle provides benefits not only in terms of power saving, but also of reduced latency
Conclusions

• In many WSN applications, traffic load varies both in time and location; these variations motivate the use of adaptive protocols at the MAC level

• B-MAC+ adapts the duty cycle of the transceiver to the traffic loads experienced by different nodes

• Easy to implement: the policy used to change the listening mode uses only local information

• No additional overhead: status information is passively disseminated among sensor nodes only when communication occurs
Future works

- Role-based characterization of algorithm’s parameters (timeout, LPL modes, number of packets)
- Better understanding of network influence (topology, node concentration)
- Use of more sophisticated statistics
- …
Thanks
(Grazie)

Any question?