Exercise Sensor Networks - (till may 30, 2005)

Lecture 5: MAC in radio networks

Exercise 5.1: Aloha with preamble sampling

A sender wants to transmit a packet to exactly one receiver via unicast (in contrast to broadcast). On the MAC layer Aloha with Preamble Sampling is implemented.

a) Why is transmitting a packet in this situation (unnecessarily) energy consuming both for the sender and the receiver?

Solution:

The longer the preamble the less idle listening is needed. But from the viewpoint of the sender a long preamble has to be sent prior to transmitting data. From the viewpoint of the receiver most nodes are woken up even though they are not addressed. Again long preambles consume energy, this time for the listeners as they have to overhear half of the preamble on average.

Exercise Sensor Networks - (till may 30, 2005)

Lecture 5: MAC in radio networks

Exercise 5.1: Aloha with preamble sampling

A sender wants to transmit a packet to exactly one receiver via unicast (in contrast to broadcast). On the MAC layer Aloha with preamble sampling is implemented.

b) How could the protocol be improved with regard to the problems identified in a) without having to synchronize the nodes? In other words: The solution should be able to work without a synchronized clock.

Solution:

The solution would be to used the preamble time for a more sensible purpose rather than only for occupying the channel as suggested by El-Hoiydi in the original work.

First suggestion:

E.g., the data packet could be sent (repeatedly) instead of the preamble. Nodes not being involved into the communication could go to sleep after having read the header of each data chunk which includes the ID of the receiver. Even the intended receiver should go to sleep after having heard the data at least once. The sender would however have to transmit the data even after the “data-preamble” since it can not know when the intended receiver woke up.

Second suggestion:

Another solution would be to fill the preamble only with the ID of the receiver. Everyone not involved could go to sleep again shortly after having woken up. If the length of the rest of the preamble was added after the ID even the intended receiver could go to sleep until the start of the data chunk.

Exercise Sensor Networks - (till may 30, 2005)

Lecture 5: MAC in radio networks

Exercise 5.2: SMAC

Ye, Heidemann and Estrin describe in their paper „An Energy-Efficient MAC-Protocol for Wireless Sensor Networks“ their SMAC approach. Therefor nodes have to synchronize themselves from time to time. Otherwise their sleep- and listen periods would diverge too much.

Some nodes may adopt more than one schedule, the one of their own cluster and one or more of neighboring clusters. The authors only describe how the schedule is synchronized within one's own cluster.

a) What is special about nodes that store more than one schedule and which know about the existence of more than one cluster, especially with regard to the synchronization?

Solution:

A node will only follow its initial schedule. The node also knowns about a neighboring cluster however the cluster might be active while the considered node sleeps. In the meantime neighboring nodes synchronize themselves at the beginning of the listening phase. But the considered nodes can not take part in this synchronization.

Exercise Sensor Networks - (till may 30, 2005)

Lecture 5: MAC in radio networks

Exercise 5.2: SMAC

b) The authors do not address the problem from a). How could it be solved?

Solution:

First suggestion

Nodes wake up at all known schedules and synchronize like they would with their initial home cluster.

+ Synchronization is not a problem
- The most important border nodes will run out of energy first because they have to be running at two active listening periods. If border-nodes fade first the sensor network gets partitioned after about 50% of its lifetime. Partitioning means that most parts of the network are lost.

Second solution

Neighboring nodes add sync information (their time to sleep) to their data packets. A receiver can sync. the schedule accordingly. This is often called piggy-backing in literature.

+ Hardly any additional overhead
- Nodes that do not often/never get packets are not synchronized. This is in particular true for sensor networks and one-way communication. Maybe in this case adding the information to the ACKs or CTS messages would mitigate the problem.

Possible solution for sparse communication: A node which has not synchronized with another cluster for a longer time could wait for a full listen- and sleep period to catch a sync packet.

Exercise Sensor Networks

Lecture 5: MAC in radio networks

Exercise 5.3: WiseMAC

A sender wants to transmit a message to a receiver using WiseMAC. Therefor it emits a preamble prior to the estimated wake-up time of the receiver and then adds the message.

a) In contrast to Aloha with preamble sampling a sender using WiseMAC knows when the receiver will wake up. What is the preamble good for in WiseMAC?

Solution:

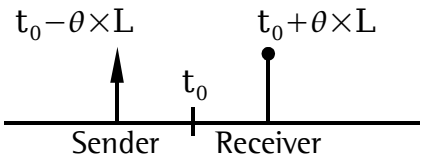
The purpose of the preamble is less focused on waking up the node as the data packet could almost be sent immediately at the known wakeup time of the recipient. A short preamble is still useful for synchronizing the beginning of the data chunk because the clocks of sender and receiver could have drifted apart. Another more important property of the small preamble is its use as contention phase if more than one sender wants to address a receiver.

b) The type of clocks being used for specific sensor nodes exhibit a maximum inaccuracy of theta time units per time unit (theta can be considered to be a small fraction, e.g., in the degree of magnitude of 10^{-5} seconds). The authors of Wise-MAC claim that after L time units a sender has to extend its preamble up to 4 x theta x L. Explain why. When does a senders have to start sending the preamble if it expects the receiver to wake up at time t_0 and if the receiver was silent for L time units?

(continued)

Exercise Sensor Networks

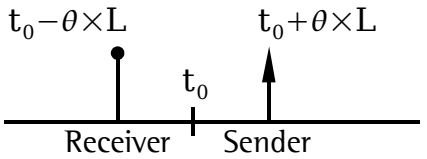
Lecture 5: MAC in radio networks



Exercise 5.3: Wise-MAC

Solution:

Case sender early, receiver late: Sender (arrow) was too fast – in fact as fast as possible so after L time units the inaccuracy accumulates to (L x theta) in the worst case. Though the sender thinks its measured time is t_0 it actually is t_0-(L x theta). The receiver was in contrast as slow as possible and advanced (L x theta) time units further than t_0. If the sender is aware that this case can happen it has to send at least (2 x L x theta) time units to reach the receiver. If this scenario was reality the sender would reach the receiver in the very last moment. To a absolutely precise the sender should even send one more bit in order to be heard by the receiver.



Case sender late, receiver early: This case is very similar to the first one, however the roles between sender and receiver are swapped. If the sender was aware of this case it knew that it would have started its preamble much too late. In order to reach the wakeup time of the receiver it would have had to go back (L x theta) to the true point of time t_0 (to account for the inaccuracy of its own clock) and another (L x theta) to account for the inaccuracy of the receiver's clock which is too fast in this case (note that a fast clock means that a node starts of listen or to send too early while a slow clock causes a note to wait too long before taking action).

Obviously, the sender can not tell whether the first or the second scenario actually occurs. So it should account for both of them at the same time. This means that it has to go back (2 x L x theta) to start as shown in the second case. Intuitively speaking we could say that the sender assumed to be much to fast and the receiver much too slow. But if both clocks were synchronous the sender would have to send 2 x L x theta to reach the receiver in the last moment. Even worse if the sender had been to fast and the receiver much to slow it would even have had to continue the preamble (4 x theta x L) time units.

Exercise Sensor Networks

Lecture 5: MAC in radio networks

Exercise 5.3: WiseMAC

A sender wants to transmit a message to a receiver using WiseMAC. Therefor it emits a preamble prior to the estimated wakeup time of the receiver and then adds the message.

c) We consider a channel which is clear at about 80% of the time and active for the rest. The occupied 20% are further subdivided into 10% preamble time and 90% time for the actual data. How long does a node have to listen who is i) the receiver of a message all the time or who is ii) always uninvolved (not addressed by a sender)? Short wake up times are not considered and we assume that the ID of the receiver is included into the message (actual data transmission phase) at the very beginning.

Solution:

Addressed nodes wakeup in the preamble phase which is no coincidence but which is planned by the sender. On average 50% of the preamble has to be overheard so that nodes are awake 0.2x0.1/2. In addition the whole data phase has to be heard of course so the addressed node listens in total:

$0.2 \times 0.1/2 + 0.2 \times 0.90 = 0.19$

An uninvolved node wakes up in the active phase with a probability of 20% whereas 10% of the active phase is preamble time. Again 50% of the preamble time has to be overheard totaling to 0.2x0.1/2. The data phase can be omitted almost entirely as the nodes realized based on the ID in the packet header that is was not addressed and goes to sleep again. With an complementary 90% the node wakes up in the data phase and has to overhear 50% of it on average. The result for the uninvolved node is

$0.2 \times 0.1/2 + 0.2 \times 0.9/2 = 0.1$

Final result: The uninvolved node is still active half as long as compared to the one being addressed.

Exercise Sensor Networks

Lecture 5: MAC in radio networks

Exercise 5.4: Hidden- and exposed station problem

Six stations are grouped around a mountain in a chain topology. Each station is able to hear the next and the previous neighbor in the chain. Station 6 and station 1 can also hear one another. Stations optimize their behavior in order to avoid collisions if possible. We consider only single packets which means that no RTS/CTS is used.

a) Station 2 is sending to station 1 already. Station 3 wants to address station 4. Is 3 allowed to send a packet and will it do so? Where does the collision occur?

Solution: 3 may send to 4. There will be a collision at station 2 and station 3. But these station do not listen but only send. The receiver 4 does however not hear 2 anymore and 1 hears 2 but not 3 anymore. In general collisions at sending stations are no problem.

b) Station 3 sends to station 2 and 5 would like to send a packet to station 4. Will station 5 start sending and should it?

Solution: Station 5 does not hear 3 and will send. At station 4 packets arrive both from 5 and 3 so that they collide. This means that station 5 should not send but can not tell.

c) Station 1 and 2 are sending. Which stations belief that they can send and which ones are actually allowed to do so?

Solution: Stations 3 and 6 know that they would possibly cause collisions somewhere and will not send. Stations 4 and 5 do not hear anything and would send, however they would also cause collisions at station 3 and 6.

Exercise Sensor Networks

Lecture 5: MAC in radio networks

Exercise 5.4: Hidden- and exposed station problem

Six stations are grouped around a mountain in a chain topology. Each station is able to hear the next and the previous neighbor in the chain. Station 6 and station 1 can also hear one another. Stations optimize their behavior in order to avoid collisions if possible. We consider only single packets which means that no RTS/CTS is used.

d) Station 1 and 4 send. Which stations belief that they can send and which ones are actually allowed to do so?

Solution: In this case every station has a sender in the direct neighborhood. So stations 2, 3, 5 and 6 believe that they must not send and they are right.

Exercise Sensor Networks

Lecture 5: MAC in radio networks

Exercise 5.5: AMRIS protocol

a) Is it possible that a msmID is used more than once in AMRIS. Why and when does it happen?

Solution:

Equal IDs can occur if stations do not hear one another due to the hidden station problem. They may even choose the same parent node not knowing about a brother node at the same level which chose the same ID. The probability of duplicate IDs in increasing with increasing distance from the root node.

b) Is it possible to address a particular node from the root even though msmIDs are not pairwise different? And is it possible for every node to address the root (we don't consider packet loss or node failure)?

Solution:

AMRIS is not a routing protocol for forwarding information for a particular node. It is used to deliver information to a group. So if a node has previously subscribed to a group it will get the information addressed to the group. The purpose of the ID is to build up a neighborhood table and to address the neighbors locally.

Note that in the worst case a nodes may have two neighbors with the same ID!

AMRIS is however suitable for finding the way from a node back to the root because everyone chose a unique parent node.