Improving Unsynchronized MAC Mechanisms in Wireless Sensor Networks

Philipp Hurni and Torsten Braun

Institute of Computer Science and Applied Mathematics, University of Bern
hurni, braun@iam.unibe.ch

Abstract. Energy-saving MAC-layer mechanisms in wireless sensor network nodes generally consist in periodic switching of a low-power wireless transceiver between an energy saving sleep mode and the costly operation modes receive and transmit. Many approaches aim to synchronize the state changes of the nodes in the network and introduce mechanisms to let the nodes synchronously wake up at designated points of time in order to exchange pending traffic. Synchronized schemes are difficult to achieve, especially over multiple hops, and introducing control messages for global or clusterwise synchronization can be a costly issue.
This paper examines improvements and optimizations on recently proposed power saving MAC protocols based on asynchronous wake-up patterns and wake-up announcements, and tests them out in a wireless sensor network integrated with an ad hoc on demand routing protocol.

1 Introduction

The design of energy efficient medium access protocols is a challenging task. It consists in finding means to use the wireless transceiver only in an *on demand* manner. In wireless sensor networks, this task is of crucial importance, as the transceiver hardware is accountable for a major part of a node's energy consumption. To save energy, the transceivers have to be switched into a low-power sleep state for a maximum amount of time, yet still maintaining connectivity to the neighboring nodes, in order to keep the network operable. The major part of many power saving mechanisms consists in introducing central or distributed synchronization and periodic switching between a sleep state and a wake state. Such synchronization measures however always cause new overhead, especially when applying multi-hop synchronization schemes. In low-traffic scenarios, periodic control message overhead can exceed the energy spent for the actual payload. Recent publications therefore proposed variants of unsynchronized power saving mechanisms.
In this paper, we lean on the basic concepts of a fully unsynchronized power saving mechanism introduced in [1] and [2], apply it to a wireless sensor network scenario and integrate principles of the WiseMAC [3] protocol. The following sections first introduce into previous work on unsynchronized power saving mechanisms, then suggest and discuss improvements concerning the wake-up

patterns and quantify the simulated efficiency gains in different networking scenarios. The novelty to be discussed lies in the introduction a moving preamble sampling period besides the fixed sampling period of WiseMAC, through which a deterministic wake-up scheme with fewer collision and fairness problems can be obtained.

2 Related Work

2.1 S-MAC

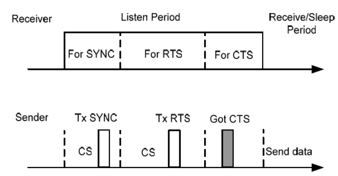

Fig. 1: S-MAC duty cycle

The S-MAC [5] protocol aims to synchronize the wake-up patterns of the nodes. It aims to let the nodes simultaneously wake up and fall back to sleep. S-MAC follows a *virtual clustering* approach to synchronize the nodes to a common wake-up scheme with a slotted structure. By regularly broadcasting SYNC packets at the beginning of a slot, neighboring nodes can adjust their clocks to the latest SYNC packet in order to correct relative clock drifts.
In a bootstrapping phase, nodes listen for incoming SYNC packets in order to join the ad-hoc network, and join a virtual synchronization cluster. When hearing no SYNC's, a node starts alternating in its wake-up pattern and propagates its schedule with SYNC messages. A problem of the virtual clustering arises when several clusters evolve. Bordering nodes in-between two clusters have to adopt the wake-patterns of both clusters, which imposes twice the duty cycles to these nodes. An S-MAC slot consists in a listen interval and a sleep interval. The listen interval is fragmented into a synchronization window to exchange SYNC messages, and a second and third window dedicated to RTS-CTS exchange. Nodes with receiving a RTS traffic announcement will clear the channel with a CTS respective window, and stay awake during the sleep phase, whereas all other nodes will go back to sleep.
The slot length and duty cycle must be set in a fixed manner, which severely restrains latency and maximal throughput. This can be disadvanteageous, as traffic can often be of bursty nature and the rate of traffic can vary over time.

2.2 T-MAC

The static duty cycle duration of S-MAC results in high latency and lower throughput, especially when varying the load level. Timeout-MAC (T-MAC)

[6] is proposed to enhance S-MAC unter variable load, and introduces an adaptive duty cycle. In T-MAC, the listen interval ends when no activation event has occurred for a given time treshold TA. An activation event may be the sensing of any communication on the radio, the end-to-end transmission of a node's data transmission, overhearing a neighbor's RTS or CTS which may announce traffic destined to itself.
One drawback of T-MACs adaptive time-out policy is that nodes often go to sleep too early. T-MAC proposes a *Future Request to Send* (FRTS) control message to alleviate this so-called early-sleeping problem, to keep neighboring nodes awake for later data exchange when having lost the contention.

2.3 WiseMAC

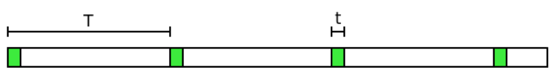

Fig. 2: WiseMAC operating with fixed cycle duration T and short medium sampling interval t

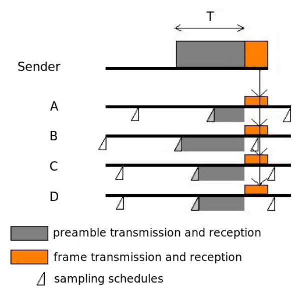

Fig. 3: WiseMAC broadcasting

The WiseMAC [3] protocol's wake-up scheme consists of simple periodic wake-up's and duty cycles of only a few percent in order to sense the carrier for a preamble signal, as depicted in Figure 2. A preamble precedes each data packet for alerting the receiving node not to go to the sleep state upon reception of the frame. As in [1], all the nodes in a network sample the medium with a common basic cycle duration T, but their offsets are independent and left unsynchronized. If a node receives the preamble signal when waking up and sampling the medium, it continues to listen until it receives a data packet or the medium becomes idle again. If a node does not know its neighbors' wake pattern yet, it sends a preamble of duration T, in order to reach the sampling interval of the neighboring node. After successful frame reception, the receiver node piggybacks its wake-up pattern in the acknowledgement message, which is then kept in a table containing the neighboring nodes' relative schedule offset from the own wake pattern. Based on this table, a node can determine the next wake-up of all its respective neigbors, and minimize the preamble length for all upcoming frames to the maximun clock drift that the two involved node's clocks may have developed during the time since the last wake pattern update.
In the WiseMAC protocol, the carrier sensing range is chosen to be larger than the transmission range in order to avoid collisions and mitigate the hidden node problem. For broadcasting, WiseMAC proposes to prepend a preamble of dura-

tion T on every broadcast message, in order to first alert every neighboring node for the upcoming transmission, and finally transmit the frame - exactly as it is done when sending a frame to a node when not knowing its sampling pattern. As illustrated in Figure 3, this broadcasting scheme uses a lot of energy only for sending and receiving the long preamble, whereas the actual data transmission may be much shorter.

2.4 Unsynchronized Power Saving Mechanism with Fixed and Random Interval

The mechanism proposed in [1] and analyzed in a static multihop wireless ad hoc network environment in [2] defines two wake and two sleep periods during one basic cycle duration T, as depicted in Figure 4.

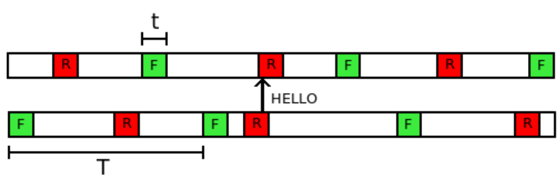

Fig. 4: Unsynchronized with Fixed and Random Interval

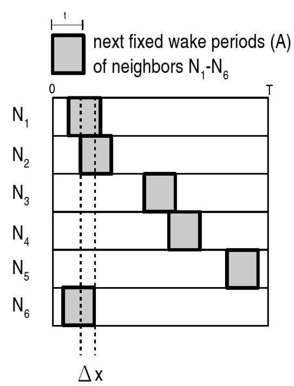

Fig. 5: Announced fixed wake periods in an intersection table

Nodes strictly alternate between a fixed wake period (F) and a random wake period (R). Each of the wake periods shall be of the same duration t. The start of the random wake period (R) is uniformly distributed between the end of the fixed wake period (F) and the start of the next one. All nodes operate with the same basic cycle duration T, although remaining unsynchronized, and switch between wake and sleep states in their individual wake-up pattern. Nodes all operate with the same wake ratio $W = 2t/T$.
The fixed wake period (F) enables a node aiming to contact any neighboring node, if its periodically occurring fixed wake period pattern is known.
However, if there is no intersection between the fixed wake periods of the sender and the neighbor, it may never learn about its presence. This motivates the choice for the random secondary wake period (R). It ensures that two nodes with disjoint wake-up pattern will sooner or later be awake at the same time and therefore be able to exchange announcements about their own wake period. By receiving these, nodes will be capable to reach all neighboring nodes during their fixed wake period (F).

As examined in [2], this wake-up scheme can easily be applied to multi-hop wireless ad hoc networks and reactive routing schemes. In order to efficiently disseminate broadcast messages, one can exploit the information about the next soonest wake-ups of each node's neighboring nodes.
By figuring out the best instant for sending and forwarding a broadcast message, [2] suggests to make optimal use of the so-called wireless multicast advantage. A node intending to broadcast a message can figure out the best instant to forward the message. The best instant shall be the instant during the next basic cycle T when the largest subset of the neighboring nodes is awake, aiming to transmit the message during some neighbor's intersections, if there are.
Figure 5 depicts the concept to search the best instant. The node calculates the best instant for broadcasting a messaget to be in-between Δx. The aim of the broadcast is therefore not to reach all of the neighbors, but only the largest possible count of neigbors with each attempt, as it is done in probabilistic broadcasting techniques, which furthermore alleviates the broadcast storm problem.
Using this technique, and taking the two best instants for rebroadcasting a route request of an on-demand routing protocol, the success ratio reached 97% even for the very low wake ratio of 4%. By rebroadcasting each message twice in every node, the disadvantage of the unsynchronized wake-pattern in regard of broadcasting becomes negligible, when considering the efforts that would otherwise be necessary to achieve a rigidly synchronized wake-pattern.

3 Simulation Environment

In all upcoming simulation scenarios, we used the OMNeT++ Network Simulator [8]. We made use of the Mobility Framework from TU Berlin [9], a framework to support simulations of wireless and mobile networks within OMNeT++. This framework incorporates a sophisticated transmission model which is based on calculation of SNR (Signal-to-Noise Ratio) and SNIR (Signal-to-Noise-and-Interference Ratio) values according to a restricted free space propagation model. This model takes transmitter power, distance, wavelength and path loss coefficient of signal dispersion into account.
The radio propagation model does not take multipath propagation or doppler effects into account, but allows to adjust the path loss coefficient α. Recent examinations of the signal attenuation in IEEE 802.11-based networks [4] conclude that a path loss coefficient between 3 and 4 is most suitable to model wireless propagation in office buildings and outdoor areas. Many sensor network simulations incorporate a path loss of 3.5 and more for wireless sensor network scenarios. We therefore sticked also to the same value of $\alpha = 3.5$. The energy consumption model is based on the amount of energy that is used by the transceiver unit. We do not take processing costs of the CPU into account. Each node's energy consumption is calculated in respect of the time and input current that the nodes spend in the respective operation modes idle/recv, transmit and sleep. Furthermore, state transition delays are incorporated to model the state

transition costs.
It is planned to later port the WiseMAC and the mechanisms described below to the Embedded Sensor Boards (ESB) of ScatterWeb [10]. The Simulation parameters are therefore tailored to model the ESB node's hardware characteristics. The ESB is equipped with a low power micro controller, various sensors, and a tr1001 transceiver module.

Simulation Parameters	
nodes	90 (uniform distribution)
area	300m $\times$ 300m
communication range	50m
carrier sensing range	100m
bitrate	19,2 Kbps
carrier frequency	868 MHz
transmitter power	0.1 mW
SNR threshold	4 dB
path-loss coefficient α	3.5
MAC & routing header	80 bit
payload	80 bit
sleep current	5 μW
transmit current	12 mW
idle/recv current	4.5 mW
recv to sleep transition delay	10 μs
recv to send transition delay	12 μs
send to recv transition delay	518 μs

4 Optimization of the WiseMAC Broadcast

With WiseMAC, broadcast transmissions must be of the duration of the sampling period to wake up and reach every node in range. As illustrated in Figure 3, this broadcasting scheme wastes a lot of energy for sending and receiving the long preamble, whereas the actual data transmission is much shorter.
In [2], this problem is studied when applying the unsynchronized wake-up pattern discussed in [1], which shares many similarities with WiseMAC. Both mechanisms propose to renounce on any global or clusterwise synchronization scheme, and only exchange information about the nodes' schedule offsets, and all nodes' wake patterns operate with a basic cycle duration T. We ported the k-best-instants heuristic of [2] into the WiseMAC mechanism of periodic preamble sampling with $T = 250ms$ and a 5% duty cycle to sense the carrier for pending traffic. An even lower duty cycle might be possible on some sensor hardware testbeds, but due to impreciseness and unpredictable behaviour of the state transitions, 5% should be an appropriate and realistic choice.
Broadcasting is of high importance when dealing with on-demand routing protocols such as AODV [7] or DSR. Nodes aiming to transmit a packet have to search a path to the destination by initiating costly Route Request floods. We

chose AODV as a well-established, efficient routing protocol, because its one-hop paradigm fits well to WiseMAC with its schedule offset table of the one-hop neighbors, and because AODV neglects to transmit and store the full routing information between two endpoints. With AODV, the route knowledge itself is distributed in the network, which makes sense in a resource-constrained wireless sensor network.
We tested out the performance of the upper schemes in an AODV route establishment scenario where every node in the network aims to find a route to the sink. In the following, the nodes first go through a neighborhood discovery process of 5 seconds during which they find their respective neighbors by sending a few HELLO messages using the original WiseMAC broadcast mechanism. After 1 minute, the first node emits a AODV route request message for the sink as it wants to start reporting data. After receiving a route response, the packet is forwarded hop by hop to the source by unicast. In intervals of 5 seconds later, one node after the other does the same, until every node has found a route to the sink. After 500s, the simulation is stopped, and the total energy consumption of all nodes calculated and summed up.

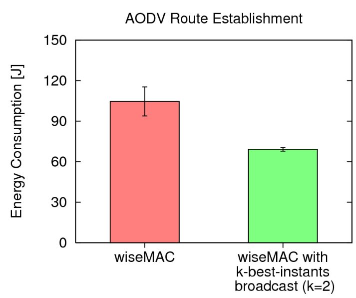

Fig. 6: Improved Broadcast scheme for WiseMAC

With both broadcasting techniques, every node managed to find a path to the sink and transmit the unicast packet. As we can see in Figure 6, the k-best-instants approach already leads to an efficiency gain of approximately 40% in this simple AODV Route Establishment scenario. The performance gain of the k-best-instants broadcasting technique weights as much as broadcasting and flooding mechanisms are used in the scenario. For example in an application scenario where queries are flooded to the nodes, the energy consumption of the approach operating with the WiseMAC broadcasting technique can be vastly improved by applying the k-best-intersections-broadcasting scheme.

5 Optimization of the WiseMAC periodic wake-up scheme

WiseMAC [3] proposes to switch the transceiver between receive and sleep state in a fixed periodic manner, and incorporating no synchronization between the

nodes other than learning each others sampling patterns. [3] argues that systematic collisions that would have been introduced through synchronization are mitigated using a probabilistic medium reservation scheme. As depicted in Figure 7 every node has its own switching pattern. Once a node has been turned on, it starts alternating between the receive and the sleep state, which leads to uniformly distributed medium samplings of the nodes. Systematic overhearing, as it occurs in synchronized MAC protocols like SMAC [5] and TMAC [6], does only seldomly occur, as in most cases, the non-intersecting wake-up intervals of the nodes naturally lead to a so-called *probabilistic overhearing avoidance*.
This scheme however has also clear drawbacks: The static deployment of a simple fixed-period sampling pattern makes it *impossible* for nodes to learn about the presence of their local neighbors just by overhearing messages originated by them. Systematic and permanent overhearing is energy waste, but some limited and infrequent overhearing can be advantageous, especially in ad hoc networks. With wake-up schedules piggybacked to all MAC frames, nodes overhearing traffic of neighbors can always update their schedule offset table. With only one fixed period wakeup pattern, nodes need to rescan the local neighborhood periodically in order to discover neighboring nodes, either by using the WiseMAC broadcasting scheme or other techniques. One simple WiseMAC broadcast right after deployment does not guarantee that all nodes were reached. It is possible that the broadcast was interferred or has failed due to bit errors.

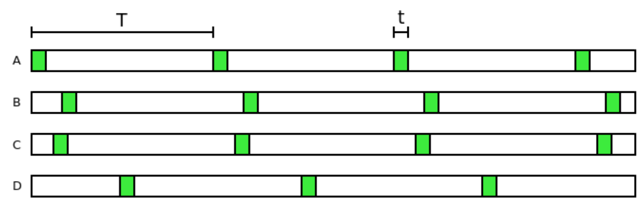

Fig. 7: WiseMAC nodes operating with fixed wake-up pattern

Furthermore, two nodes with nearly identical sampling patterns might systematically hinder each other from receiving messages destined to them. Consider node B and C in Figure 7, which share almost the same wakeup pattern. Assume that all node are at least in interference rage of each other.
Node C always slightly precedes the wakeup period of node B. If two respective neighbors A, D want to reach B and C, the transmission $A \rightarrow B$ will always be shadowed by the transmission $D \rightarrow C$, as node C always wakes up earlier. D will always be capable of sending the preamble and start transmitting the frame to C, whereas B will wake up, notice that there is a transmission going on that is not destined to itself and go back to the sleep state after the medium is idle again. A will have to wait until there is no message transfer to C such that it can finally transmit to B. This leads to a high latency for A's packets whenever there is traffic destined to C.
Such problems can have severe impact on the service properties for a large part of the nodes, especially if C and D are neuralgic spots in the sensor network

which have to forward data packets from whole subtrees.
Another drawback of the single periodic wake period occurs when applying the k-best-instants broadcasting scheme introduced in [2] to on demand route request querying, as the neighboring nodes will always consider the same nodes' intersections for rebroadcasting a frame and therefore stick to the same behaviour in every retry attempt. This is especially the case when there are bottlenecks in the network topology and is investigated furthermore in [2].
The WiseMAC fixed periodic wake-up mechanism can be improved in a quite simple manner. We can achieve a medium reservation scheme with similar properties but a better probabilistic overhearing avoidance and better medium utilization by keeping a fixed wake-up period and integrating a moving wake period in between two fixed wake periods. A node then strictly alternates between a fixed wake period and a moving wake period, similar to the mechanism proposed in [1].

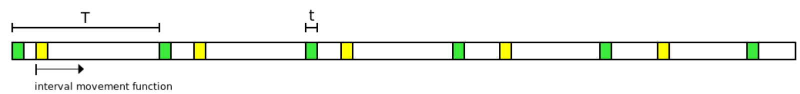

Fig. 8: fixed wake period (green) and moving wake period (yellow)

[1] proposed the choice of a fixed wake-up pattern and a random wake-up period in between to solve the problem that nearby nodes are possibly never detected due to non-intersecting wake patterns. We suggest a mechanism with the fixed wake period and the moving wake period, as shown in Figure 8, for the following reasons:

- The behaviour of the moving wake period is deterministic rather than random and follows a simple linear movement function, which is identical and predictable in every node. If a node needs to transmit a message to one of its neighbor, it first checks whether the neighbor's next fixed wake period or the next moving wake period is soonest.
  Using previously received wake-pattern annuncements, the node then determines the next soonest wake period of the neighbor, prepones a preamble to the frame and then awaits its neighbor's wake-up exactly as in WiseMAC.

- The problem that non-intersecting wake-up patterns could lead to nodes never discovering each other is also resolved. Sooner or later, nodes will overhear frames or acknowledgements, even from or to nodes with non-intersecting wake-up patterns. The moving wake interval ensures that - given some periodic low-rate traffic - this will happen within a limited amount of time.

- Using this wakeup pattern, the upper problem with the concurrent transmission between the nodes $A \rightarrow B$ and $D \rightarrow C$ is solved in an elegant manner. Consider the situation depicted in Figure 9:
  Node A aims to transmit a frame to node B at the instant indicated with the

upper arrow (i.e. it may have to forward the frame after receiving it during its own fixed wake period), and D aims for transmission of a frame to C at the instant indicated with the lower arrow. D will find the medium idle at the start of the next fixed wake period of node C and will transmit the packet. A will find the medium busy at the start of the next fixed wake period of B, and not access the channel. It will be able to wait for the next moving wake period of B, which does not intersect with the respective moving wake nor fixed wake period of C.
The movement function of the moving wake period leads to a floating of the node's wake periods over time. If there was only a fixed period, A would have to wait until D has no more packets to send. If D continously generates or forwards packets, the traffic that needs to be forwarded by B is blocked. This can lead to high delays for packets forwarded by B, causes fairness problems and can even lead to buffer overflows at B.

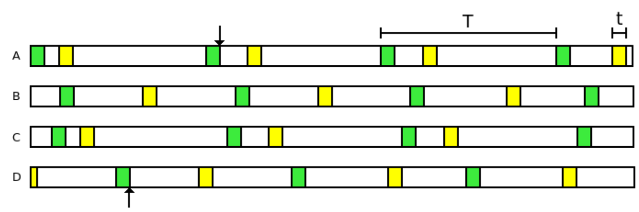

Fig. 9: Problem of concurrent transmissions

5.1 Node-to-Sink Periodic Traffic Scenario

We compared the approaches of the k-best-instants WiseMAC Scheme (original wakeup pattern but improved broadcasting technique) with the approach of the moving wake intervals described above. In case of the WiseMAC wake-up pattern, we chose $250ms$ as interval between two duty cycles. In case of the moving wake interval approach, we chose $500ms$ as wake-up interval between two fixed duty cycles. Like this, the expected value of the wake-up interval of both approaches equals 250ms, and the service characteristics of both approaches can be compared.
We chose the same networking setup as in the upper scenario, and let every node report data starting at $t = 60s$ with poisson traffic of increasing rate λ during $1h$. As we are only dealing with the route establishment in the beginning and the broadcasting scheme is the same in both approaches, the comparison mainly covers the properties of the unicast node-to-node acknowledged datagram service from the sources to the sink when applying the different wake-up schemes.

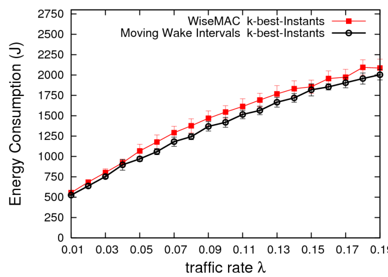

Fig. 10: Energy Consumption

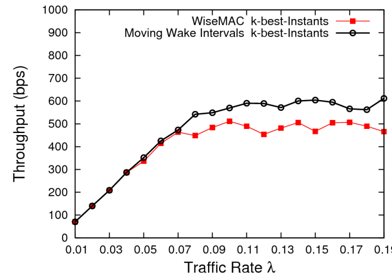

Fig. 11: Throughput

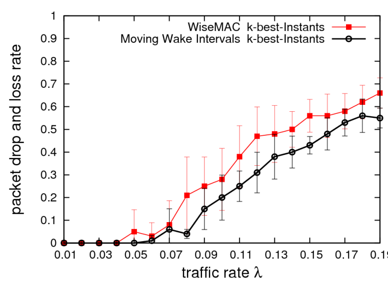

Fig. 12: Packet Drops and Losses

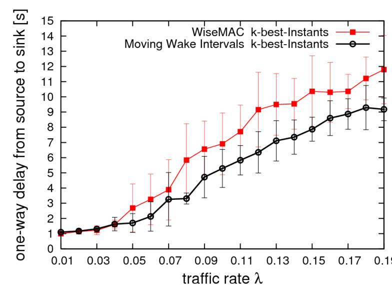

Fig. 13: One-Way-Delay

The moving wake interval approach leads to a slightly lower energy consumption and a better throughput with increasing traffic rate. The performance gains in regard of throughput and energy consumption are measurable, though remain below 20%.

It is insightful that the mechanism with the moving wake periods performs better only with increasing traffic. As long as there is not much traffic, the situation with the concurrent transmissions described above does not or only seldomly occur. With increasing traffic, congestion problems arise earlier with the fixed static wake-up pattern of WiseMAC.

5.2 Distributed Events Scenario

Similar results as in 5.1 can be observed in a distributed event scenario. We triggered events with poisson rate λ on a random position point on the same uniformly distributed topology. When a event happens at a certain position on the $300m \times 300m$ plane, each node in the vicinity of $50m$ starts reporting data with between 1 and 3 packets.

The results also show an improvement of the moving wake period mechanism in comparison with the fixed wake pattern, which is slowly increasing with increasing event rate.

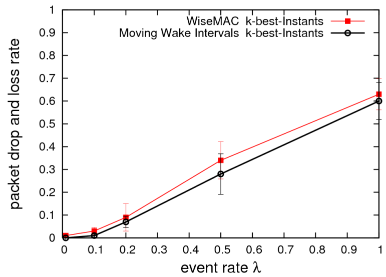

Fig. 14: Packet Drops and Losses

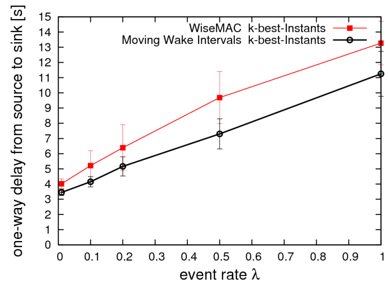

Fig. 15: One-Way-Delay

6 Conclusions

This paper combines features and ideas of previous work on unsynchronized MAC protocols for wireless sensor networks and finds performance optimizations in regard of energy efficiency, throughput, fairness and latency. It shows that mechanisms suggested in [1] and [2] can be applied to improve the yet very effective WiseMAC [3] power saving MAC protocol. Porting the k-best-instants broadcasting technique to a multihop wireless sensor network led to performance gains in comparison with the WiseMAC broadcasting scheme. We showed that the WiseMAC fixed periodic wake-up scheme can cause fairness problems with increasing traffic, which can be alleviated by adapting the wake-up scheme to a hybrid scheme with a fixed periodic wake-up and a moving wake-up.

References

1. Braun, T., Feeney, L. M.: *Power Saving in Wireless Ad hoc Networks without Synchronization*, 5th Scandinavian Workshop on Wireless Ad-hoc Networks, 2005
2. Hurni, Ph., Braun, T., Feeney, L.M.: *Simulation and Evaluation of Unsynchronized Power Saving Mechanisms in Wireless Ad Hoc Networks*, WWIC 2006
3. El-Hoiydi, A., Decotignie, J.-D.: *WiseMAC: An Ultra Low Power MAC Protocol for Multihop Wireless Sensor Networks*, ALGOSENSORS 2004
4. Faria, D.: *Modeling Signal Attenuation in IEEE 802.11 Wireless LANs* Technical, Stanford University, July 2005.
5. Ye, W., Heidemann, J., Estrin, D.: *An Energy Efficient MAC protocol for Wireless Sensor Networks.* 21st International Conference of the IEEE Computer and Communications Societies (Infocom), 2002.
6. Dam, T. V., Langendoen, K.: *An Adaptive Energy Efficient MAC Protocol for Wireless Sensor Networks.* ACM Conference on Embedded Network Sensor Systems (SenSys), 2003.
7. Perkins, Ch., Belding-Royer, E.: *Ad hoc On-Demand Distance Vector (AODV) Routing*, IETF Internet draft RFC 3561, October 2003
8. Varga, A.: *The OMNeT++ Discrete Event Simulation System (http://www.omnetpp.org)*,
9. Mobility Framework for OMNeT++ (http://mobility-fw.sourceforge.net)
10. Scatterweb: *Platform for wireless sensor networks* (http://www.scatterweb.net)