# MERLIN: A Synergetic Integration of MAC and Routing Protocol for Distributed Sensor Networks

A.G. Ruzzelli, G.M.P O'Hare, M.J. O'Grady, R. Tynan
Adaptive Information Cluster (AIC)
School of Computer Science and Informatics, University College Dublin, Ireland
{ruzzelli, gregory.ohare,michael.j.ogrady, richard.tynan}@ucd.ie

***Abstract*— Notoriously, energy-efficient MAC protocols cause high latency of packets. Such delays may well increase when a routing protocol is applied. Therefore, quantifying the end-to-end delay and energy consumption when low duty cycle MAC and routing protocols are jointly used, is of particular interest. In this paper, we present a comprehensive evaluation of the MERLIN (MAC and Efficient Routing integrated with support for localization) protocol. MERLIN integrates MAC and routing features into a single architecture. In contrast to many sensor network protocols, it employs a multicast upstream and multicast downstream approach to relaying packets to and from the gateway. Simultaneous reception and transmission errors are notified by using asynchronous burst ACK and negative burst ACK. A division of the network into timezones, together with an appropriate scheduling policy, enables the routing of packets to the closest gateway. An evaluation of MERLIN has been conducted through simulation, against both the SMAC and the ESR routing protocols,which is an improved version of the DSR algorithm. The results illustrate how both SMAC and ESR, jointly used in low duty cycle scenarios, can cause an impractical and very high end-to-end delays. MERLIN, as an integrated approach, notably reduces the latency, resulting in nodes that can operate in a very low duty cycle. Consequently, an extension of the operative lifetime of the sensor network is achieved.**

***Index Terms*— wireless, sensor, networks, MAC, routing, protocol, energy-efficient self-organizing, multicast.**

## I. Introduction

Distributed Wireless Sensor Networks (WSNs) comprise large numbers of wireless devices deployed over a physical environment that actively cooperate to accomplish one or more tasks. Sensors are designed to operate unattended over long durations, frequently in hostile environments. A sensing component forms an essential part of the device; popular examples include temperature, accelerometer, humidity, infrared light, pressure, and magnetic sensors, as well as chemical sensors. Many potential applications of WSNs may be found in the literature, for example, environment monitoring [11], intelligent buildings [20], and object tracking [4]. Further novel applications envisage collaboration of sensors and Radio Frequency ID (RFID) tags with mobile devices, for example in the field of logistics [4] and for military operations in [8]. Certain features are desirable in WSNs, including robustness, dynamic programmability and low unit cost per sensor. In the latter instance, this invariably results in simple architectures, low processing capabilities and low memories. Though potential application domains for WSNs are wide and varied, certain characteristics are desirable in almost all circumstances. These include efficient use of energy over the WSN's lifetime, robust and reliable communications, scalability in terms of network size, and dynamic network adaptivity in response to changes in the prevailing operating conditions. When these design characteristics are considered, it can be seen that satisfactorily addressing a random WSN application poses significant difficulties: difficulties that differ significantly from those encountered with wireless ad-hoc networks. Energy efficiency, signal robustness as well as the ubiquitous cost issue, all raise particular issues that must be adequately addressed if the WSN is to fulfil its objectives.

In this paper, current developments with MERLIN (Mac and Energy-efficient Routing with Localization support INtegrated), an integrated architecture for distributed sensor networks, are described. These developments build on earlier work in [17] resulting in the introduction of a novel transmission mechanism. A simulation of MERLIN is presented along with a comprehensive evaluation of its performance in comparison to SMAC [23] and the Eyes Source Routing [22] (ESR) protocols, as such protocols provide a useful benchmark against which MERLIN can be compared. In particular, ESR, as the standard routing assessed and used in the EYES project, is an improved version of the well known Dynamic Source Routing [7] (DSR) protocol.

## II. Motivation

Recent studies on transmission radius, for example [24], have demonstrated how protocols that are theoretically effective may perform poorly when deployed in realistic environments. Interference, multi-path effect and fading may cause a premature deterioration of the signal. In addition, nodes are not reliable and can often fail due to low cost hardware thus leading to a dramatic reduction of probability of receiving packets correctly. Furthermore, better performances in terms of latency are sacrificed for an extension of the network's operational lifetime. The result is a time delay that can exceed one second per hop, as shown in [23]. Time delay due to the MAC protocol is not the only relevant issue as it is also expected that end-to-end delay would increase when a routing protocol is applied. Hence, it is interesting to quantify the end-to-end delay when low duty cycle MAC and routing protocols are jointly used. Whether the total delay might be acceptable or not will be dictated both by the application domain requirements and network size.

Usually, energy efficiency and latency are addressed separately, from an access control or effective routing prospective. The protocol MERLIN adopts a different approach in that it embraces the energy efficiency and communication reliability issues by merging information commonly addressed in different layers into the same protocol architecture. It is envisaged that better performance of sensor networks can be achieved through a pragmatic integration of MAC and routing protocol into a single architecture. The advantages are: (1) A reduction of the end-to-end packet latency, thus preserving competitive energy efficiency for mutual node/gate communication; (2) An increase of the probability of packet reception at the gateway by providing controlled packet duplication to address sensor failure and bad channel condition; (3) A more efficient usage of resources such as the memory capacity and processing capability of the sensor. However, MERLIN is not optimized for communication between sensor nodes as peers located several hops away in different parts of the network. Rather, the objective of MERLIN is to provide efficient and reliable communication for sensor network applications.

## III. Related work

After a decade of research on wireless sensor networks, the scientific literature offers a profusion of energy-efficient protocols. Among them, SMAC [23] is probably the most well known and the one that was adopted for the assessment as it is often used as a benchmark for other architectures. In SMAC, a node alternates periods of activity to period of sleeping. The sleeping period is further divided in three contention based access sections: the SYNC period dedicated to node synchronization update and the Request To Send (RTS) period and the CTS period. Nodes periodically send a SYNC packet to synchronize the beginning of the active period with the neighbourhood. A node that has data to transmit firstly contends the channel for the transmission of a SYNC packet and then follows the RTS/CTS/DATA/ACK handshake mechanism. Idle listening is high because even if no packets have been transmitted, nodes keep on listening until the end of the RTS section, or until they detect an RTS message not addressed. All packets include the duration of the transmission so that neighbouring nodes can set up a NAV timer and go into sleep so refraining from transmitting. Finally, the adaptive listening technique allow neighbouring nodes to take up passing the message a further hop away thus reducing the latency.

Improvements of SMAC in terms of energy consumption were achieved by TMAC [2], directly derived from SMAC, that forces nodes to start transmitting at the beginning of the active periods. Both protocols have large end-to-end latency and idle listening at the receiver.

An interesting MAC protocol for WSNs is BMAC [14] that combines the mechanisms of low power listening [3] [6] and clear channel assessment techniques to reduce the idle listening at the receiver. A drawback of BMAC is the poor performance under high contention scenario because of the usage of the wake-up preamble. In fact, in case of high channel contention, the wake-up preamble may be transmitted consecutively by several neighbouring nodes so preventing transmission of the actual data. Furthermore, SMAC, TMAC and BMAC use RTS/CTS, hence they present high overhead due to small sensor data packets. Recently, Rhee et al. developed Z-MAC [16] that outperforms BMAC in high contention scenario by a combination of TDMA and CSMA access but with the drawback of higher energy consumption for low contention scenarios. A similar architecture to MERLIN is DMAC [10] which incorporates a data gathering tree to reduce the latency. The drawback of this technique is that it is suitable only for unidirectional communication flow to a single gateway.

With respect to the routing protocol, the DSR [7] and AODV [13] algorithms are the most cited due to their flexibility and light weight that match properly the resource limitation of the tiny sensors. For this evaluation of MERLIN, we used the Eyes Source Routing [22] (ESR) derived from the two previous architectures. ESR has been demonstrated to have better energy consumption profile than DSR. The above mentioned protocols differ from MERLIN as they all address MAC and routing issues separately.

## IV. Deployment considerations

Before delving into the architecture of MERLIN, it is prudent to consider some pertinent issues relating to sensor network deployment. An essential requirement is the presence of at least one gateway as a ”special” node of the network that collects data broadcast from the distributed sensors. As such, it is preferable for the gateway to have augmented energy and processing capabilities. In the case of a large scale sensor network, it is advisable to deploy multiple gateways. For a better coverage of the network, it is advisable to place gateways appropriately so that there is a minimum disparity between sizes of subnetworks covered by each gateway. In essence, this is a network topology problem, which is beyond the scope of this paper.

In cases where multiple gateways are deployed, MERLIN requires that their clocks are synchronised, possibly by means of an alternative communication system like satellite, WLAN or WMAN. This enables the initialisation process as described in section V-D. Furthermore, previous study in [5] showed that MERLIN can support node location through the usage of a Received Signal Strength Indicator (RSSI) device, usually incorporated into standard off-the-shelf sensors.

## V. The design of MERLIN

Fundamental to the MERLIN protocol is the timezone concept, in which the sensor network is subdivided into time zones as illustrated in 1. Time zone division commences during the initialisation phase with the broadcasting of an initial SYNC packet from the gateway to neighbouring nodes which, in turn, will augment the SYNC packet before forwarding it to surrounding nodes. At the end of the initialisation phase, all nodes will be organised into appropriate time zones. In contrast to the message exchange scheme adopted by other network protocols, nodes in MERLIN do not nominate a specific neighbouring node to forward its packet. Rather, it

employs *multicast upstream* and *multicast downstream* to relay information to the gateway and away from it as required. Furthermore, synchronization, as well as timezone updates, are all exchanged by means of a periodic local broadcast. Asynchronous burst ACK and negative burst ACK indicate successful reception and errors respectively. Controlled multi-path is employed as a means of reducing packet redundancy and ultimately transmission overhead. Periodical node activity is regulated by the node's scheduling table.

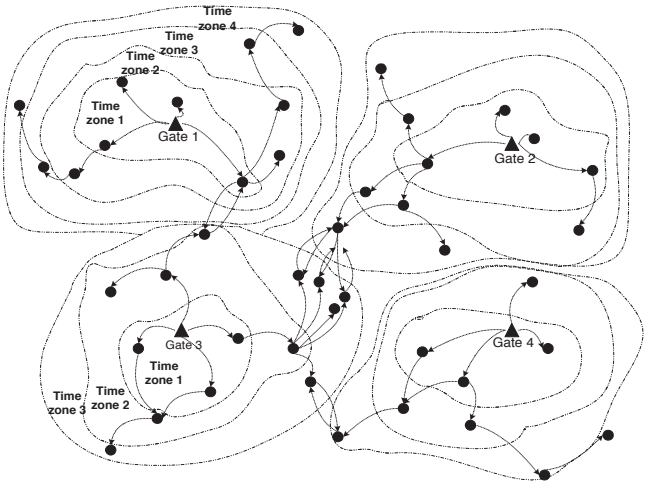


Fig. 1. Division of the network in timezones following the simultaneous network flooding by gateways of a SYNC packet. Every node sets its zone and forward the packet to more distant nodes.

### A. Data traffic

MERLIN adopts a multicast transmission approach for communicating packets to adjacent time zones. In particular, it is optimised for communications between nodes and gateways and vice versa. Two types of multicast are supported: (1) **Upstream multicast** towards the gateway; (2) **Downstream multicast** away from the gateway. Furthermore, MERLIN provides a **local broadcast** data traffic that is used for local exchange of information among neighbouring nodes (e.g. RSS indicator, table of neighbours, battery level, localization info etc.). In the case of downstream multicast and local broadcast, the notification of at least one incorrect reception is sufficient to reschedule packet retransmission.

### B. Scheduling

The purpose of the scheduling table is to allocate time-slots to the nodes in the network to assign periods of node activity and inactivity. The scheduling allows synchronizing neighbouring nodes for transmission and reception. In MERLIN, *nodes in the same timezone use the same slot to transmit*. The scheduling table is usually transmitted by the gateway during the initialisation phase. Further work in [19] shows how further tables can be opportunistically and dynamically be injected into the network as application requirements dictate.

Figure 2 shows the scheduling table adopted by MERLIN referred to as V-table (the name V-table is due to the V-shape communication flow). The length of the table is equal to the length of 1 frametime and comprises 4 timezones. Each small rectangle represents a slot. When a timeslot is allocated for transmission to a particular timezone, the adjacent zone owns the slot for reception while nodes in further timezones are in sleeping mode. Note that the V-table performs fast upstream and downstream multicasts by forwarding a packet to 4 timezones towards the gateway or in the opposite direction within the same frametime. The allocation of further zones can be obtained by appending the same table. Furthermore, scheduling for further frames is obtained by flanking the same table.

The 4-zone V-table allows potential parallel transmission of nodes that are located 4 zones apart. Appending the scheduling table for further zone transmission shows that an increase of the number of zones in a table results in a less timezone parallel transmissions. Therefore, it is clear that a smaller number of zones in a table results more parallel zone transmissions. Although the schedule table might be further reduced to a 3-zone table, the irregularity of the timezones due to random node location caused significant collisions at the zone in between parallel transmission. Our empirical results in [18] proved for the 4-zone V-Table to be the most advantageous number that caused high number of parallel retransmission together with minimum number of collisions.

The last column of timeslots in the V-table is dedicated to local broadcast. In order to avoid inter-zone collisions due to simultaneous local broadcast by nodes the following formula applies:

$$Mod(NFRAME, 4) = Mod(NZONE, 4)$$

Where $NFRAME$ is the incremental number of frames common to all nodes that is cyclically repeated from 0 to 100 and $NZONE$ is the node timezone. In this way nodes in the same timezone can contend the slot for local broadcast only once each 4 frametimes. Although not depicted in figure 2, it is intended that when a zone is scheduled for local broadcast, nodes within the same zone and adjacent ones are in listening mode.

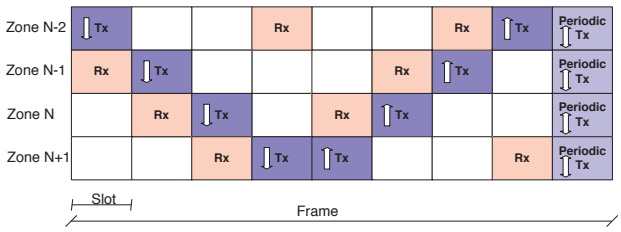


Fig. 2. The table of scheduling with periodic local broadcast

### C. Transmission mechanism

Recall that in MERLIN, time is subdivided into slots as described in the scheduling section in V-B. All time slots are provided with a *contention period* (**CP**), located at the beginning of the time slot, as illustrated in figure 3. The cases of receiving and transmitting nodes are now considered:

**Receiver Nodes:** MERLIN adopts the *clear channel assessment* (CCA) mechanism through *Low Power Listening* (LPL),

an approach that is effectively utilized in BMAC [14] and deployed in TinyOS motes. With the LPL technique, the radio wakes-up and samples the channel for a short time period of 4ms [14] referred to as CCA period. In MERLIN, nodes scheduled for reception activate the radio for a CCA period to listen to the channel. If there is no activity in the channel, the node goes into sleep mode; otherwise it receives the packet. As a result of adopting the CCA mechanism, idle listening time at the receiver is significantly reduced.

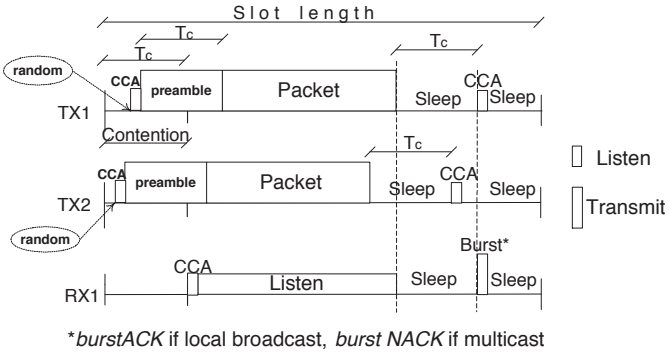


Fig. 3. Transmission mechanism for collision avoidance of MERLIN

**Transmitter Nodes:** As shown in figure 3, a node that wishes to transmit initially chooses a random time within the contention period and wakes up at that time to sense the channel for a CCA period. If nothing is detected then the channel is assumed free of carriers and the node immediately commences transmitting the packet preamble, the duration of which is **Tc** equal to whole CP length. In this way, the preamble transmission is guaranteed to reach the end of the CP. The data packet is transmitted immediately after the preamble.

The random starting time allows asynchronous transmissions that are utilized by MERLIN to notify multiple correct receptions by means of short **burstACK**, referred to as *BACK*. We stress the fact that such bursts do not carry any coded information and multiple overlapping bursts are identified at the receiver as single burst. For *upstream multicast transmission, one burstACK is sufficient to notify the receiver of correct reception of at least one node in the zone closer to the gateway*. In fact, when multicasting upstream, the transmitter does not need to know all nodes participating to the forwarding process. Rather, the notification of at least one correct reception is enough to consider the packet forwarded. The scheduling table ensures that the packet has been transmitted in the correct direction, in fact, in the case of upstream multicast, only nodes in one time zone lower have their radio in listening mode due to the scheduling scheme. This is not applicable for both local broadcast and downstream multicast which may carry information for all neighbours or from the gateway to the network respectively.

In case of downstream multicast or local broadcast transmission, which is identified by dedicated slots in the scheduling table in figure 2, the burst tone identifies a reception error. This is referred to as **negative-burstACK** (*BNACK*). In fact, a local broadcast is intended to be received by all neighbouring nodes. In case of BNACK reception the transmitter reschedules the packet. A shortcoming of the BNACK mechanism is that a node that is momentarily unreachable (e.g. if a vehicle is passing by) might not be aware of a packet transmission hence it does not send back a BNACK. The problem is alleviated if the transmitter identifies packets with an incremental ID and make a backup of the last few packets sent. Packets can be uniquely identified by a combination of node ID and incremental packet ID. If a further transmission occurs, the receiver can identify the missing packets by a mismatch in the incremental ID number hence requests a copy of them. However, the packet recovery procedure described results in an increase of transmission delay and the number of backups is limited by the memory constraints of the device.

In essence, there is not difference in the signal transmitted by the two burst tones. BACK and BNACK can be simply identified by means of the slot in which the transmission occurs. BACKs are transmitted in slots dedicated to upstream multicast while BNACKs are transmitted in downstream multicast and local broadcast. In both multicast and local broadcast, if an error is detected, the packet will be rescheduled after a random exponential back-off procedure. The burst tone is usually provided in the standard sensor radio stack. So as to be detectable, the burst transmission is delayed by exactly `Tc` after the end of the maximum packet length allowed. A primary consequence of this is that the possibility of accidentally corrupting nearby transmissions is eliminated. In addition, the burst transmission is kept asynchronous, thus ensuring that the transmitter can return to the sleep state after the transmission is complete, and can wake-up after `Tc` to detect either a BACK or BNACK related to its transmitted packet.

### *D. Initialisation*

Nodes start the network initialisation phase by listening for a **SYNC packet** that contains *timing information*, *sender ID* and *sender timezone*. Gateways, synchronized among them, start initialising the network by broadcasting SYNC packets. Gateways are considered to be in timezone 0. Sensor nodes in the vicinity of the gateway that receive the SYNC packet will use it to synchronize their internal clock; they are now classified as belonging to the timezone 1. As depicted in figure 1, timezone 1 nodes will then forward the SYNC packet to more distant nodes. The forwarding activity is accomplished by means of the transmission mechanism for collision avoidance described in section V-C. All sensors receiving SYNC packets from nodes in timezone 1 will be classified as belonging to timezone 2. This procedure is repeated until all nodes are assigned a timezone. In the case of multiple gateways, the gateways start flooding the network simultaneously by sending a SYNC packet to neighbouring nodes which are then set in timezone 1. Subsequently, the SYNC packet is forwarded to more distant nodes which are then set in timezone 2. In this way, nodes can approximate their timezone with respect to the closest gateway. At the end of the initialisation phase, *the timezone number is equal to the number of minimum hops a packet needs to reach the closest gateway*.

During the initialisation, it is possible that a node is momentarily classified in a higher timezone. For example, should a node in a timezone N fail to access the channel, or if its packet collides, the node can re-apply the channel contention procedure the successive frame. In the meantime, any node waiting for initialisation can receive a SYNC packet through a third party, possibly causing it being temporally assigned to a higher timezone. Such a temporal timezone assignment does not prevent the node from communicating with neighbouring nodes, though this may result in a longer route to the gateway. This situation is rectified when the node receives the timezone N SYNC packet resulting in a path with a fewer number of hops to the gateway. In the case of nodes being equidistant from two gateways the node will select a timezone according to the first SYNC received. The total initialisation time depends on the scheduling table being utilized. In particular, previous studies on the performance of the scheduling of MERLIN in [18] showed that the V-table, described in section V-B, enabled the initialisation of 14 timezones in a network in approximately 9 seconds.

### *E. Synchronization*

As a result of the poor hardware sophistication node synchronization needs to be repeated periodically by the gateway. In addition to synchronizing the starting moment (offset) of the slots among nodes, the frequency skew of each node's individual clock needs to be compensated for continuously. Both of these issues are solved by including the time of transmission on each radio transmission. All receiver nodes can then estimate the start of the slot according to the sender, and synchronize their clock offset. Furthermore, the clock skew can be estimated by comparing over time the observed difference between a sender's clock and the node's own clock. In MERLIN, *nodes update their clock synchronization from the nodes belonging to the lower timezone* as they are closer to the gateway. In fact, such nodes are considered to hold a finer synchronization; the procedure is much alike the stratums used in NTP [12]. With respect to scalability, new nodes can join the network by simply listening to any packet. The information contained such as sender timezone, packet transmission time and type (e.g. upstream multicast) allow the new node estimating its position and synchronization. The node is then in position to join the network.

If a new gateway wants to join the network, firstly it has to join it as a new node then it announces its presence through local broadcast of a SYNC packet. Nodes will compare the old timezone number with the new one and forward the SYNC packet. In case of a new closer gateway, they will change their timezone at the beginning of the successive frame only if the transmission has been successful.

### *F. Packet format*

This section addresses packet and message formatting issues. In MERLIN a packet is essentially a collection of messages that is assembled when a node is required to forward a number of messages received from other nodes. A message is generated by a node and contains protocol info and sensed data such as sourceID, destID, forwardID, forwardZone, msgID, msgType, and, of course, the DATA payload. All messages are uniquely identified by the msgID, which is a combination of sourceID and private message number ID.

In the process of forwarding packets to the gateway, a node can receive messages from different nodes to be forwarded to nodes on lower time zones. Since sensor data are usually few bytes, for example, temperature, pressure, chemical data and so on, MERLIN can concatenate the messages and then transmit them as a packet. Therefore, a node that has more than one message to send, it will send them in one packet in the same time slot thus saving energy and reducing overhead in transmission. The maximum number of messages in a packet is dictated by a number of factors such as application type, transceiver data rate, and channel condition. For example, a packet should be kept small if being transmitted in a very lossy channel. In the MERLIN standard configuration, the packet size does not exceed 100 bytes (maximum packet length) although for simulation purposes, a larger range of packet sizes were considered.

In order to facilitate the concatenation process, MERLIN provides three small buffers for local broadcast, upstream multicast and downstream multicast. Such buffers are used to temporarily store a node's own messages as well as messages received from other nodes for transmission. Every time a packet of a certain type is received, all messages contained within it are compared with those in the appropriate buffer and any duplicate messages are deleted. The buffer is then updated with the new set of messages. Packets for transmission are formed by following the FIFO principle. The transmission process of the different type of packets is regulated by the scheduling table, described in section V-B.

### *G. Routing characteristics*

The division in timezones together with the described scheduling and type of data traffic allow packets to be routed to and away from the closest gateway. Recall that MERLIN does not address a specific forwarding node. This may cause duplication of packets during forwading activity. However, the packet generation is controlled through a mechanism of overhearing. Furthermore, this simple but effective routing is improved by a mechanism of on demand zone maintenance.

*1) Controlled multi-path:* As stated before, messages can be concatenated to form a packet as described in section V-F, when a packet is formed, an msg-index is generated. The msg-index contains all msgIDs of messages within the packet. The msg-index, or the msgID in the case of a single message, is located at the beginning of the packet, thus allowing neighbouring nodes of the same timezone overhear what messages are being transmitted. A dividing code can allow identifying the beginning of the msg-index. On identifying messages in its buffer that have already been transmitted by a neighbouring node in the same time zone, such messages will be immediately deleted from the buffer as shown in figure 4. This mechanism is used for multicast upstream communications as it is only necessary that one instance of the message reaches the gateway. Conversely, in a downstream multicast scenario,

this mechanism cannot be applied as deleting a message may result in some nodes not receiving communication from the gateway, for example, periodical network updates.

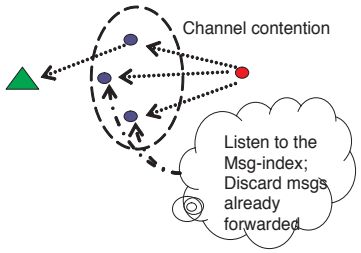


Fig. 4. The controlled multi-path mechanism through neighbouring nodes overhearing

*2) Timezone maintenance:* Should a node that is located in zone N not receive the periodic timezone update from any neighbouring node in the zone N-1, it may decide to transmit a timezone update request (TUR) through upstream multicast. In case of no answer, the node will assume that the connection with the zone N-1 failed. The node will try to re-establish connection with any node within the same timezone through local broadcast of TUR. Should the node receive at least a burstACK, it will change its timezone to N+1. Otherwise, the node will assume a failed connection also with nodes within the same zone. Hence, it will try to re-establish a connection through a downstream multicast transmission of TUR. At this stage, a reception of a burstACK means a change of timezone to N+2. For all cases, TUR transmission is repeated twice before assuming a failed route connection.

## VI. Simulation

All simulations of MERLIN have been conducted within the OmNet++ [15] discrete event simulator. The sensor network framework of OmNet++ is based upon earlier work on the co-design template [21], used in the EU EYES project [15], and slightly modified to obtain performance metric values for this assessment. This section describes comprehensively the performance metrics, scenarios and parameters used for the assessment so that they can be used for replication of the experiments by other researchers. Consequently, the simulated results are presented and discussed.

### A. Scenarios and Performance Metrics

The metrics used for the assessment of MERLIN against both SMAC and SMAC+ESR are the following:

*a) Energy consumption per message received:* the amount of energy required by a node to relay a message from source to destination.

*b) Network lifetime:* the operative network life expectancy under network parameters described in section VI-B. The network is considered to ”die” when 30% of nodes are depleted since some parts of the network might be disconnected to the gateway;

*c) End-to-end latency:* the time that elapses between the source node transmission and the destination node reception, also referred to as EtoE latency;

*d) Total message overhead:* the total number of messages generated by the network activity including control messages and duplication of data messages due to multiple path generation;

*e) Percentage of sleeping time:* the percentage of node sleeping time calculated with respect to the total node activity time;

Node depletion is connected to the periodical wake up concept through the node duty cycle which is the ratio between the listen and sleeping activities in a single time period. Hence, the assessment shows both network lifetime and EtoE latency of packets with respect to the node duty cycle. The node energy consumption has been calculated by computing the duration of sleeping, transmitting, receiving and switching between states. In fact, as demonstrated in [1], the switching energy becomes a significant source of energy consumption for low duty cycle such as 10% or less.

MERLIN is assessed under two network scenarios: a 5 node two-hop scenario and a random multihop scenario. For consistency and fairness in the evaluation of MERLIN against SMAC, scenarios and setup parameters are equal to those used in the assessment of SMAC in [23]. For multihop scenarios in particular, SMAC uses the ESR protocol to relay data to the gateway. Furthermore, protocol scalability has been evaluated under both high and low conditions of data traffic and network density.

### B. Simulation setup

The simulation setup was configured using parameters obtained from the Transceiver Tr1001 data sheet in table VI-B, as well as with measurements of the switching energy [1] obtained directly from the EYES node [9]. All experiments, with the exception of one on the transmission overhead, used data message of 16 bytes as this is considered a typical size of sensor data payload, for example sensed temperature data. However, for the total packet overhead, it is considered appropriate to show the behaviour for a wider range of packet size for a number of simulation of 10 minutes each. Data traffic has been studied under both a 12 and 60 messages per minute generation rate, referred to as low and high data traffic respectively. In line with the initial SMAC assessment in [23], its duty cycle is obtained by dividing the listen period by the sleep period. The SMAC periodic listen time has been kept constant at 80ms so node duty cycle is modified by changing the SMAC sleeping period length. In MERLIN, similar duty cycle can be achieved by dividing the CCA length by the frametime with the observation that a node wakes up twice in a single frametime to transmit upstream and downstream, and once every 4 frames for local broadcast. Therefore, similar duty cycle configurations could be achieved by changing the frametime length. Both protocols schedule a synchronization period every 13s, by means of local broadcasts by SMAC and gateway flooding by MERLIN, to compensate for node clock skew and offset. With respect to the ESR, we kept the same setup used in the earlier evaluation of the protocol against DSR [7], as described in [22].

The 5 nodes two-hop scenario in figure 5, which is the same used to assess SMAC in [23], consists of 2 source

| current [mA] | | | power [mW] | | |
|---|---|---|---|---|---|
| Sl | Rx | Tx | Sl | Rx | Tx |
| 0.005 | 4.8 | 12 | 0.015 | 14.4 | 21 |
| **switching time [$\mu$s]** | | | | | |
| Sl to Rx | Sl to Tx | Rx to Sl | Tx to Sl | Rx to Tx | Tx to Rx |
| 700 | 700 | 10 | 10 | 700 | 700 |
| **switching energy [$\mu$J]** | | | | | |
| 8.82 | 25.2 | 0.116 | 2.83 | 25.2 | 8.85 |

TABLE I

DATA SHEET OF THE EYES PROTOTYPE TRANSCEIVER AND DIRECT MEASUREMENTS OF SWITCHING PARAMETERS

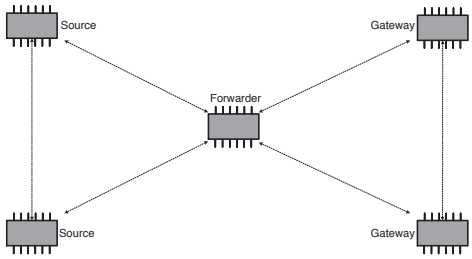


Fig. 5. Local scenario of 5 nodes, 2 sources, 1 forwarder, 2 destinations

nodes, 1 forwarding node and 2 gateways. The chosen signal strength causes contention for both source nodes to transmit as well as for both gateways to send an acknowledgment. The forwarder is within the transmitting range of both sources and gateways. Measurements of the energy consumption per message forwarded to the gateway have been conducted on the node forwarder as this is considered the one most subjected to node activity and collisions. Message transmission starts after an initialisation time of 20s and ends when 100 packets are successfully forwarded to one of the gateways.

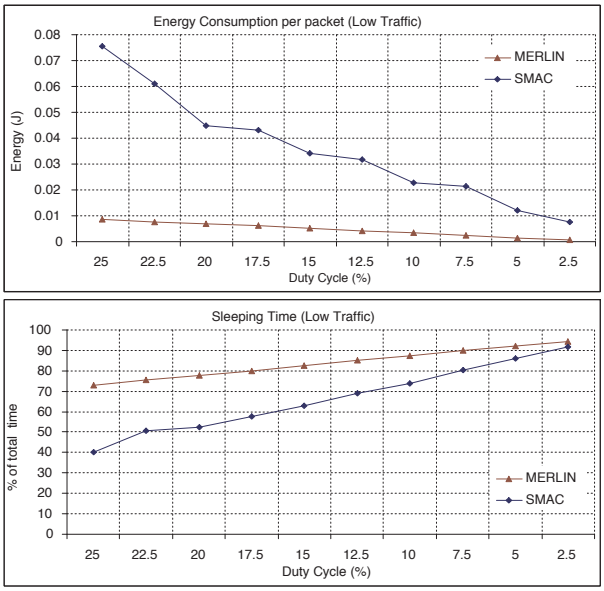


Fig. 6. Energy consumption and sleeping time in low data traffic condition for two-hop scenario.

The multi-hop scenario consists of 70 nodes randomly placed in the network. Random scenarios have been obtained by using 10 independent seeds for each run. Identical numbers but different seeds have been used to randomly select the source nodes for generating data. Periodically, five nodes are arbitrarily selected to report a data packet to the gateway. Results are averaged from the obtained measurements. The network lifetime has been calculated based on a 2J battery and assuming linear energy depletion. In order to evaluate the scalability of MERLIN, the multi-hop scenario comprises high and low node densities, obtained by keeping the transmitting radius constant and selecting two different network sizes of $400 * 300m^2$ and $600 * 500m^2$ respectively. Recall that experiments have been conducted under combinations of both high and low data traffic and network density.

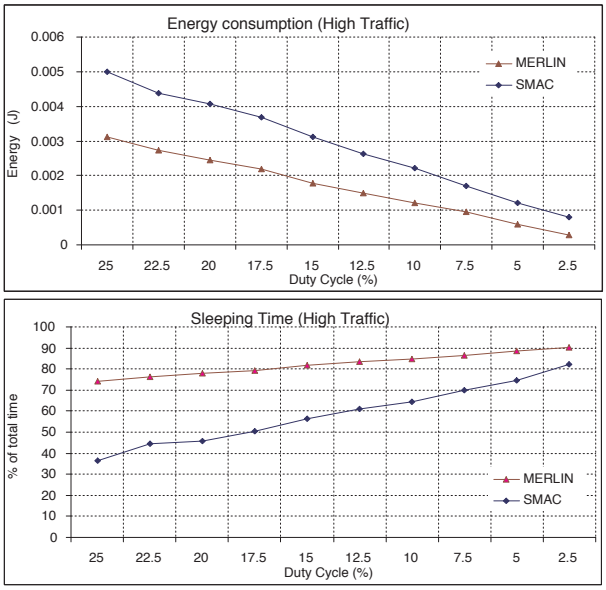


Fig. 7. Energy consumption and sleeping time in high data traffic condition for two-hop scenario.

## C. Simulation results

**Two-hop scenario:** The graphs in figure 6 show the energy consumption and sleeping percentage per message forwarded to the gateway for node 2 under low data traffic conditions with respect to the node duty cycle. Graphs show a quasi-linear behaviour of energy consumption that decreases as the duty cycle is reduced. The results show a better performance of MERLIN for the entire range of studied duty cycles. As duty cycle percentage increases, so does the performance difference between the two protocols. In fact, the node in SMAC wakes up more often to listen to the whole SYNC and RTS periods. This causes a large increase in idle listening time at the receiver. The result is confirmed by the graph of the node sleeping time percentage in 6.

It is interesting to compare the low traffic graphs in figure 6 with the high data traffic graphs in figure 7. The comparison of energy consumption in SMAC shows that the low data traffic conditions leads to an even higher energy consumption,

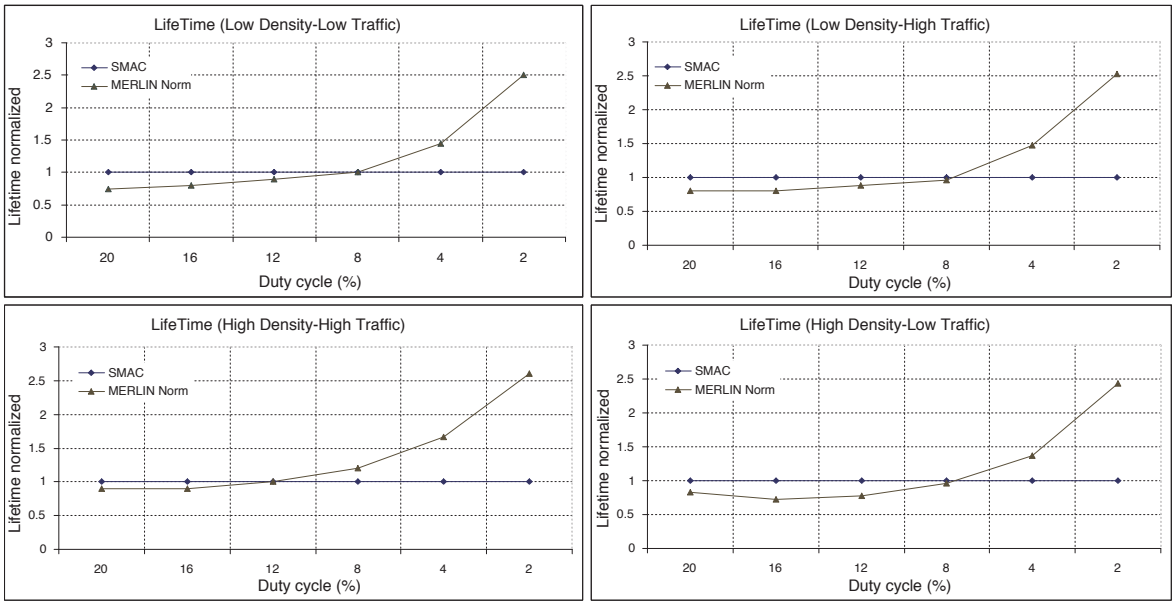


Fig. 8. Comparison of network lifetime for different combination of node density and data traffic.

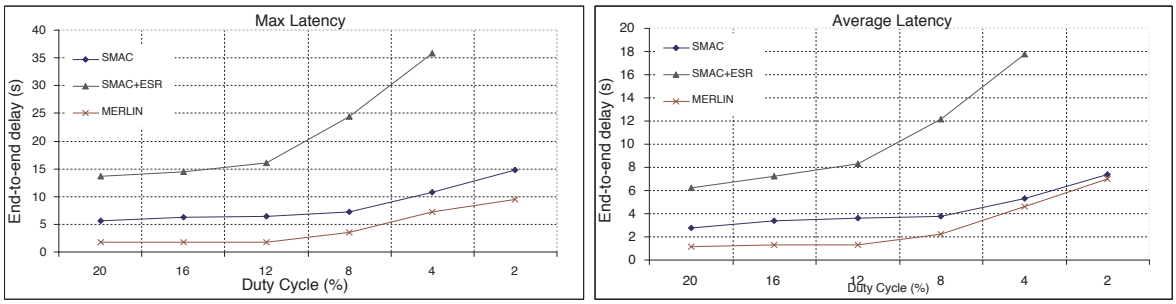


Fig. 9. Comparison of EtoE latency for MERLIN, SMAC and SMAC+ESR in the multihop scenario

when contrasted with high traffic conditions. This is because energy consumption is calculated over the timespan that is required to receive 100 message correctly. Therefore total simulation time is much longer for low data traffic than for high data traffic. Specifically in SMAC, energy consumption is mainly dominated by node idle listening, which is one of the main limitations of the protocol. In contrast, the combination of scheduling and a short CCA effectively minimise node idle listening in MERLIN. Both traffic settings present large energy saving in MERLIN with respect to SMAC. Energy saving increases for higher duty cycles, particularly in low data traffic scenarios. The main reason for this is the high idle listening time of SMAC that is more pronounced when a node wakes up more frequently and when data traffic is light. This characteristic is confirmed by the graphs of node sleeping percentage in figures 6 and 7 respectively.

**Multi-hop scenario:** Graphs in figure 8 show the behaviour of SMAC and MERLIN in terms of network lifetime under a wide range of duty cycles for the scenario described in section VI-A. At first glance, it should be noticed how each graph shows a value of duty cycle which corresponds to a crossover point of the two studied protocols. In particular, MERLIN is more energy efficient than SMAC after the crossover point for smaller values of node duty cycles. However, such graphs have a limited relevance if the end-to-end (EtoE) latency of packets is not considered. In fact, the energy consumption, dictated by the application, is closely related to the message delay from source to destination. For example, applications that sustain a delay of n seconds present a very different network lifetime profile to applications that sustain a delay of `2n` seconds even if the same protocol is used. To this end, graphs in figure 9 illustrate the behaviour of both maximum and average EtoE latency of message for MERLIN, SMAC and SMAC + ESR. In particular, the maximum EtoE latency is due to communication from those nodes most distant from the gateway, which are approximately 11 or 12 hops away from it. Note that the highest threshold of latency of SMAC+ESR was set at 40s for the simulations of the maximum latency. Therefore, in the case of 2% duty cycle, the maximum latency could not be depicted.

Based on the graphs, the following observations may be made:

1) In terms of maximum EtoE latency, MERLIN outperforms SMAC between 1.5 and 3 times and SMAC+RSR

between 3.7 and 5 times.

2) In terms of average EtoE latency, MERLIN outperforms SMAC by about 20% but performance tends to converge as duty cycle is reduced; MERLIN outperforms SMAC+ESR by approximately a factor of 4 and 5.
3) The simulation demonstrates that reducing the duty cycle causes an approximate exponential increase of EtoE latency for SMAC+ESR.

By jointly analysing both the energy consumption and EtoE latency results, the previous graphs of the network lifetime can be interpreted differently. In fact, if a 10s or higher threshold of maximum EtoE latency is sustained by the application, MERLIN will be set up at 2% duty cycle thus resulting in an extension of the network lifetime 2.5 times higher than SMAC. In the case of a sustained maximum threshold of EtoE latency less than 10s, the protocol couple SMAC+ESR is unsuitable. In contrast, MERLIN can still be used until an EtoE latency of 1.8s is reached, by increasing the duty cycle. As a result, by setting both SMAC and MERLIN at their optimum according to a certain maximum EtoE delay requirement of the application, MERLIN always presents a longer network lifetime than SMAC.

We conclude our assessment by looking at the total overhead in transmission, defined in section VI-A generated by MERLIN and SMAC+ESR. In particular, we evaluated the total message overhead of the combination of SMAC for channel access and ESR for routing against the access to the channel and controlled multiple path generation of MERLIN. As earlier described in section VI-B, the total overhead has been evaluated in a number of random multihop topologies for wide range of message lengths in condition of high data traffic and high density as considered the worst case scenario.

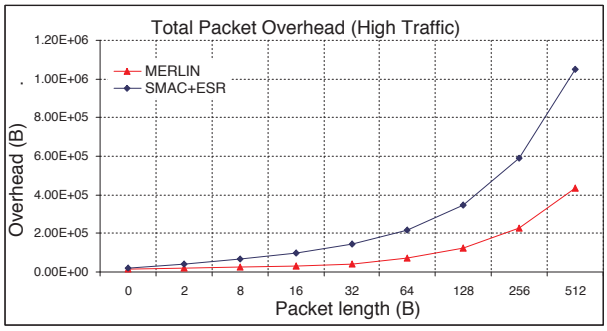


Fig. 10. Comparison of total transmission overhead for MERLIN and SMAC+ESR

The combination SMAC and ESR necessitates a large amount of control messages such as RTS, CTS, periodic SYNC and periodic route maintenance packets that have a great impact on the total transmission overhead. The situation can be improved by working on their coordination, for example by merging in one packet the periodic SYNC and route maintenance. This is not an easy task due to separate protocol architectures hence different protocol needs. In MERLIN, the integration of MAC and routing allow cutting the number of control packets. This in turn compensate for the duplicate packets caused by multiple paths that are generated by forwarding activity. Figure 10 shows large saving of packets transmitted by MERLIN with respect to the couple SMAC+ESR for the whole range of packet size simulated.

## VII. Conclusion

Normally, protocols for wireless sensor networks present a high end-to-end latency profile sacrificed for energy saving. Such a delay is expected to increase when a combination of energy-efficient MAC and routing are jointly used thus making protocols unsuitable for certain applications. In this paper we presented a comprehensive description and evaluation of MERLIN as an integrated architecture for sensor networks. MERLIN is suitable for communication to and from gateways and nodes. The protocol is not optimized for communication between nodes as peers located several hops distant. The protocol has been assessed against the combination of Sensor-MAC and ESR routing. The evaluation showed how in multihop node/gateway communication the MERLIN integrated architecture leads to both significant energy saving and latency reduction with respect to a wide range of node duty cycle. Finally, it was demonstrated that if the duty cycle of SMAC+ESR was reduced to 10% or less, an end-to-end delay of several tens of seconds was incurred. In contrast, MERLIN remained stable under a 10 second delay until 2% duty cycle. Hence the energy consumption profile of MERLIN results in a network lifetime being extended by a factor of 2.5 with respect to SMAC.

## VIII. Acknowledgments

Gregory O'Hare, Antonio Ruzzelli and Richard Tynan gratefully acknowledges the support of Science Foundation Ireland under Grant No. 03z/IN.3/1361. Michael O'Grady gratefully acknowledges the support of the Irish Research Council for Science, Engineering & Technology (IRCSET) though the Embark Initiative postdoctoral fellowship programme. Special thanks from Antonio Ruzzelli to Kamin Whitehouse for the useful exchange of emails that revealed some initial issues that have later been solved.

## References

[1] G.M.P. O'Hare R.Tynan A.G. Ruzzelli, P. Cotan and P.J.M havinga, *Protocol assessment issues in low duty cycle sensor networks: The switching energy*, In proc. of IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing ,Taiwan, June 5-7 (2006).

[2] T. V. Dam and K. Langendoen, *An adaptive energy efficient mac protocol for wireless sensor networks*, ACM Sensys (2003).

[3] Amre El-Hoiydi, *Spatial tdma and csma with preamble sampling for low power ad hoc wireless sensor networks*, In proceeding of Seventh International Symposium on Computers and Communications (ISCC'02) **685** (2002).

[4] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, *Next century challenges: Scalable coordination in sensor networks*, Mobile Computing and Networking, 1999, pp. 263–270.

[5] L. Evers, W. Bach, D. Dam, M. Jonker, J. Scholten, and P. J. M. Havinga, *An iterative quality-based localization algorithm for ad hoc networks*, 1st Int. Conf. on Pervasive Computing (Pervasive), Zurich, Switzerland (2002), 55–61.

[6] J. Hill and D. Culler, *A wireless embedded sensor architecture for system-level optimization*, 2001.

[7] D. B. Johnson and D. A. Maltz., *Dinamic source routing in ad hoc wireless networks*, Mobile Computing **353** (1996), Kluwer.

[8] A. Ledeczi, A. Nadas, P. Volgyesi, G. Balogh, B. Kusy, J. Sallai, G. Pap, S. Dora, K. Molnar, M. Maroti, and G. Simon, *Countersniper system for urban warfare*, ACM Transaction on Sensor Networks **1** (2005), no. 2, 153–177.

[9] P.J.M Havinga H.J. Kip L.F.W. van Hoesel, S. Dulman, *Design of a low-power testbed for wireless sensor networks and verification*, Tech. report, University of Twente and Nedap N.V., The Netherlands, May 2003.

[10] G. Lu, B. Krishnamachari, and C. S. Raghavendra, *An adaptive energy-efficient and low-latency mac for data gathering in sensor networks*, In proc. of International workshop on Algorithms for Wireless, Mobile, ad Hoc Sensor Networks (2004).

[11] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, *Wireless sensor networks for habitat monitoring*, In proc. of the International Workshop on Wireless Sensor Networks and Applications. Atlanta (2002).

[12] D.L. Mills, Z. Yang, and T.A. Marsland, *Internet time synchronization. the network time protocol.in global states and time in distributed system*, IEEE Computer society press, 1994.

[13] C.E. Perkins and E.M. Royer, *Ad-hoc on-demand distance vector routing*, WMCSA '99: Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications (Washington, DC, USA), 1999, p. 90.

[14] J. Polastre, J. Hill, and D. Culler, *Versatile low power media access for wireless sensor networks*, (2004), 95–107.

[15] EYES Project, *State of the art*, European Union, http//:eyes.eu.org, 2002-2005.

[16] I. Rhee, A. Warrier, M. Aia, and J. Min, *Z-mac: a hybrid mac for wireless sensor networks*, In proc. of the 3rd international conference on Embedded networked sensor systems, Sensys05 (New York, USA), ACM Press, 2005, pp. 90–101.

[17] A.G. Ruzzelli, L. Evers, S. Dulman, L.W. Van Hoesel, and P.J.M. Havinga, *On the design of an energy-efficient low-latency integrated protocol for distributed mobile sensor networks*, In proc. of International Workshop on Wireless Ad hoc Networks (2004).

[18] A.G. Ruzzelli, M.J. O'Grady, G.M.P. O'Hare, and R. Tynan, *An energy-efficient and low-latency routing protocol for wireless sensor networks*, In proc. of the Advanced Industrial Conference on Wireless Technologies SENET 2005, Montreal, Canada, IEEE Press, (2005.).

[19] A.G. Ruzzelli, R. Tynan, and G.M.P. OHare, *Adaptive scheduling in wireless sensor networks*, In proc. of WAC2005, the 2nd Workshop on Autonomic Communication Athens, Greece, LNCS press (2005).

[20] D. Snoonian, *Smart buildings*, vol. 40(8), IEEE Spectrum, 2003.

[21] EYES WSN, *The codesign project*, 2003, http://wwwhome.cs.utwente.nl/ mader/Codesign/.

[22] J. Wu, P. Havinga, S. Dulman, and T. Nieberg, *Eyes source routing protocol for wireless sensor networks*, Proc. of European Workshop on Wireless Sensor Networks EWSN (2004).

[23] W. Ye, J. Heidemann, and D. Estrin, *Medium access control with coordinated adaptive sleeping for wireless sensor networks*, IEEE/ACM Transactions on Networking **3** (2004).

[24] G. Zhou, T. He, S. Krishnamurthy, and J.A. Stankovic, *Impact of radio irregularity on wireless sensor networks*, In Proc. of the 2nd international conference on Mobile systems, applications, and services (MobiSys04) (New York, USA), ACM Press, 2004.