

Experimental Evaluation of An Adaptive Staggered Sleep Protocol for Wireless Sensor Networks

Giuseppe Anastasi^{*}, Monica Castronuovo^{*}, Marco Conti[#], Mario Di Francesco^{*}

Pervasive Computing & Networking Lab. (PerLab)

**Dept. of Information Engineering
University of Pisa, Italy
{firstname.lastname}@iet.unipi.it*

*#CNR-IIT
National Research Council, Italy
{firstname.lastname}@iit.cnr.it*

Abstract

In the last years wireless sensor networks (WSN) have emerged as an enabling technology for a wide range of applications. The main challenge in the deployment and actual utilization of WSNs is the scarce energy budget available at sensor nodes. In this paper we address this problem through an Adaptive Staggered Sleep Protocol (ASLEEP) which is suitable to environmental monitoring applications. By tuning dynamically the wakeup period of each sensor node to its current traffic pattern, ASLEEP reduces both energy consumption and message latency. In addition, it can adapt to changes in the operating conditions. We present an experimental evaluation of ASLEEP based on a prototype implementation in a real testbed. The experimental results show that the proposed solution provides better performance, in terms of reduced energy consumption and message latency, in comparison with other similar approaches.

1. Introduction

In the last years wireless sensor networks (WSNs) [1] have emerged as an enabling technology for a wide range of real-life applications. Among them, environmental monitoring can particularly benefit from WSNs due to low costs and fine-grained measurements, which cannot be otherwise obtained [5], [8].

In a typical scenario, a large number of tiny sensor nodes monitor the quantity of interest over a sensing field. Each node can acquire physical data (e.g., temperature, humidity, and so on), process them locally and send the results to one or more collection points (usually referred to as sinks) by using wireless

communication. As sensor nodes are battery powered, their energy budget is very limited. Thus they should embed mechanisms to reduce energy consumption.

Several approaches have been proposed to improve the energy efficiency in WSNs [11]. If we break down the energy consumption in a typical sensor node [7], we can see that the highest contribution is due to wireless communication. In addition, the energy expenditure is approximately the same when the radio is transmitting, receiving, or idle. On the contrary, a substantial reduction of the energy consumption of the radio – i.e. a few orders of magnitude – can be achieved by switching it down to a low power (sleep) mode. In this case, nodes operate on the basis of a duty-cycle, i.e. they alternate between the active and the sleep states with a given pattern. Obviously, a sleep scheduling protocol is needed to coordinate nodes so that neighbors can still communicate together.

As messages can be received by the destination only when it is awake, duty-cycling mechanisms introduce additional delays which can be significant if sleep schedules are not properly arranged [3]. In addition, static schemes (such as [6] and [9]) require to set the operating parameters (i.e. the sleep/wakeup duration) before the deployment. In this case, a proper setting is not always trivial as it would require an a-priori characterization of the network (e.g. topology, traffic etc.). Furthermore, if the operating conditions change, these schemes cannot react to such variations and, thus, have non-optimal performance in terms of energy consumption. To overcome this problem, many adaptive duty-cycle schemes have been proposed, such as [2], [10], [16] and [17].

In this paper we refer to the Adaptive Staggered Sleep Protocol (ASLEEP) defined in [10], and present an experimental evaluation of the protocol based on a prototype implementation in a real testbed. ASLEEP is targeted to periodic data collection applications, such as environmental monitoring, and can adapt to changes in the operating conditions without requiring any a-

Work funded partially by the IST program of the European Commission under the FP6-2005-NEST-PATH MEMORY project, and partially by the Italian Ministry for Education and Scientific Research (MIUR) under the FIRB ArtDeco project.

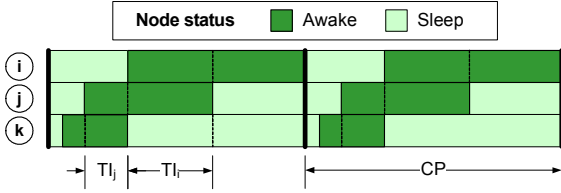


Figure 1. Sleep scheduling parameters

priori knowledge of network topology and traffic pattern. By adjusting dynamically the wakeup period of each single sensor node (through local measurements of the network activity), ASLEEP tries to minimize both energy consumption and message latency. In addition, it is implemented on top of the MAC protocol and can, thus, be used with any sensor platform.

The paper is organized as follows. Section 2 briefly describes the ASLEEP protocol. Section 3 presents the experimental setup used for its implementation and performance evaluation. Section 4 discusses the results obtained from the experiments. Finally, Section 5 concludes the paper.

2. ASLEEP protocol description

In this section we only provide a very brief description of the ASLEEP protocol. The reader can refer to [10] for details. In our scheme, sensor nodes are assumed to form a logical *routing tree* (rooted at the sink) for data forwarding. The communication between a parent and its children occurs in *communication periods* (CPs) that repeat periodically. Each CP is divided into two portions: a *talk interval* (TI), during which nodes communicate by using the underlying MAC protocol, and a *silence interval* (SI) during which nodes are sleeping.

Each talk interval is shared between nodes with a parent-child relationship. Sensor nodes span their activity over two adjacent talk intervals¹, the first with their children and the other one with their parent (see Figure 1). Throughout, we will refer to the talk interval shared by a generic node j and all its children as TI_j^m , referring to the m -th communication period CP^m .

TIs are staggered so that nodes at lower levels in the routing tree wake up earlier than their ancestors (see Figure 1). This optimizes the data communication from leaves to the root (sink). Each parent node dynamically estimates the duration of the TI to be shared with its children in the next CP, according to the algorithm described in Appendix A. Although nodes can

¹ Clearly the sink only has the talk interval with its children, while leaves only have the talk interval with their parent.

independently set their TI, a collective effort is needed for the schedule of the whole network to remain consistent and energy efficient. Hence, as a result of a change in the TI of a single node, the network-wide schedule needs to be re-arranged.

Sleep schedule re-arrangement is accomplished by appropriately shifting the TIs of nodes, so as to ensure that all TIs remain contiguous and properly staggered. Special messages, called *beacons*, are used for propagating schedule parameters to nodes in the routing tree. During regular operations, *direct beacons* are broadcasted by a parent node to all its children at each CP. Special *reverse beacons* are sent in the opposite direction – i.e., from a child to its parent – when there is a TI increase.

Direct beacons are critical for the algorithm correctness. If they are lost nodes cannot know the schedule for the next CP and, hence, they cannot communicate until a fresh schedule information is re-acquired. To add robustness to the direct beacon transmission, the algorithm provides different mechanisms for improving reliability. First, a portion of the TI – called *Beacon Period* – is reserved only for the direct beacon transmission. Direct beacons are transmitted twice with a random backoff delay uniformly distributed within the Beacon Period to reduce the chance of collisions. Second, a schedule prediction mechanism is used when a beacon is missed. When a node does not receive the expected direct beacon, it assumes the schedule parameters for the next CP are the same as the current one. In this case, nodes operate correctly, provided that there has not been a schedule change in the meantime. Otherwise, they can detect the error and recover to the actual schedule at the expense of an increased energy consumption.

3. Experimental Setup

To evaluate the performance of the ASLEEP protocol, we carried out a set of experiments in a real deployment. To this end we used a set of 15 Tmote Sky [12] sensor nodes with the TinyOS (version 1.1.15) operating system [13]. We implemented the ASLEEP protocol on top of the default CSMA/CA MAC protocol shipped with TinyOS. We configured the MAC layer to use acknowledgements, with the maximum number of retransmissions set to 5. Before starting the ASLEEP protocol, we performed an initial time synchronization by using a modified version of the synchronization protocol in [14]. We also used *MintRoute* [15] to build the routing tree. Sensor nodes were placed at about 50 cm from the ground; they periodically sampled the external temperature and

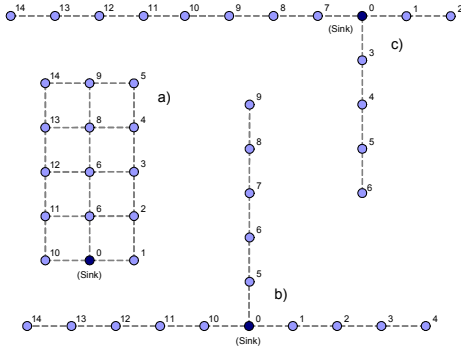


Figure 2. Scenarios: all-in-range (a), multi-hop balanced (b), and multi-hop unbalanced (c)

reported acquired data to the sink (node 0) which was connected to a laptop for data collection and analysis.

For comparison purposes, we considered the following additional sleep scheduling schemes.

- *Always-on*. In this scheme there is no sleep scheduling: nodes never go to sleep and forward messages as soon as they receive them.
- *Fixed staggered (TAG) scheme*. Sensor nodes use a staggered scheme for sleep coordination, where the TI is fixed and equal for all sensor nodes. In detail, the TI is set to the value of the CP divided by the depth of the tree, as in TAG [4].
- *Optimal fixed staggered scheme*. In this scheme the TI is fixed and equal for all nodes, as in the fixed staggered scheme above. However, the TI is set to the estimate provided by our ASLEEP protocol.

To compare ASLEEP against the above schemes, we considered the following performance indices.

- *Average Duty Cycle*, defined as the average fraction of time the one-hop neighbors of the sink remain awake. Since all traffic originated by source nodes traverses these nodes, they determine the network lifetime.
- *Delivery ratio*, defined as the ratio between the number of messages successfully received by the sink and the total number of messages generated by all sensor nodes.
- *Average message latency*, defined as the average time between the message transmission at the source node and the reception of the same message at the sink node.

In the experimental analysis we focused on the stationary behavior of the protocol in the different scenarios. We used the parameter settings shown in Table 1. We replicated each experiment 5 times (each replica was 100 communication-period long). The results presented below show the average values and

Table 1. Operational parameters

| Parameter | Value |
|---------------------------------|----------|
| Talk interval slot (q) | 150 ms |
| Beacon Period | 60 ms |
| Communication period | 15 s |
| Message size (payload) | 28 bytes |
| Window size (L) | 10 |
| TI increase threshold (g_1) | 150 ms |
| TI decrease threshold (g_2) | 300 ms |

the standard deviations over the entire set of replicas².

We started our experimental analysis by investigating the performance of the different sleep/wakeup schemes in the following three scenarios.

- *All-in-Range Scenario*. All nodes are within the transmission range of node 0 which acts as the sink. Specifically, as shown in Figure 2-a, sensor nodes are deployed on a 5 by 3 grid, at a distance of approximately 10 m, and use the maximum allowed transmission power. In theory, nodes should form a star topology in this scenario, i.e., they all should become children of the sink in the routing tree formation and communicate directly to it.
- *Multi-hop Balanced Scenario*. Sensor nodes are deployed to form a T-shaped topology with approximately even branches, as illustrated in Figure 2-b. We set the transmission power of sensor nodes to -20 dBm and measured a transmission range of approximately 12 m. Since the distance between neighboring nodes deployed along the same row is about 10 m, in theory we should expect a multi-hop topology with routing sub-trees having approximately the same depth.
- *Multi-hop Unbalanced Scenario*. Sensor nodes are deployed again to form a T-shaped topology, but sub-trees have now different depths (2, 4, and 8, as shown in Figure 2-c). Transmission power and distances between nodes are the same as in the previous scenario. Therefore, in theory, nodes should form an unbalanced multi-hop topology.

4. Experimental Results

In the following, we will first evaluate the performance of the different sleep/wakeup schemes in the previously introduced scenarios. Then, we will focus on a single scenario and evaluate the impact of different data-related parameters.

² To limit the duration of experiments, we considered a CP of 15 seconds. However, we carried out additional experiments which assess the effectiveness of our scheme even for CPs up to 4 minutes.

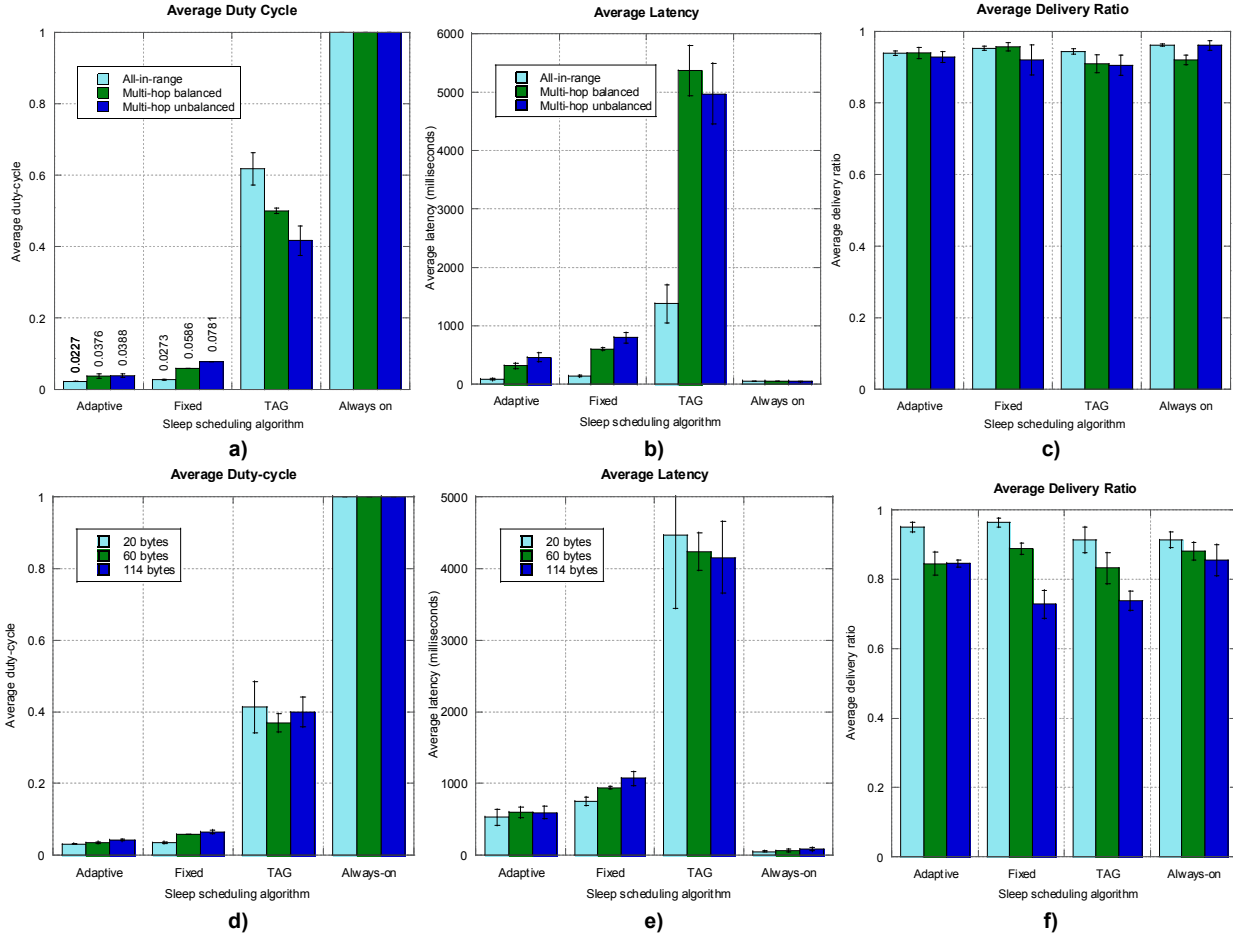


Figure 3. Performance indices for different scenarios (first row) and message size (second row)

4.1 Impact of the network scenario

In this section we analyze the performance of the different sleep/wakeup schemes in the three scenarios under investigation.

Figure 3-a compares the average duty-cycle allowed by the different duty-cycling schemes. We can see that the ASLEEP protocol allows the lower duty-cycle in all scenarios. This happens because it can match the amount of time actually needed for communication, while the other schemes impose the same activity to nodes which actually have different demands (i.e. nodes belonging to the same level of the routing tree). In addition, with the adaptive and the fixed staggered schemes sensor nodes experience higher duty-cycles when passing from the all-in-range, to the multi-hop balanced, to the multi-hop unbalanced scenario. Clearly the difference between the delivery ratio obtained with the adaptive and the fixed staggered schemes increases as well, so that in the multi-hop unbalanced scenario the adaptive scheme consumes approximately a half

than the fixed staggered one. Conversely, for TAG³ the trend is the just the opposite, as in this scheme the length of the TI depends on the inverse of the routing tree depth, and overall obtains very high values.

Figure 3-b shows the average message latency introduced by the different duty-cycling schemes under the three scenarios. Also in this case the adaptive scheme is better than the other solutions, obviously except for the always-on scheme. This is strictly due to the fact that the adaptive algorithm gets shorter activity times, so that less time elapses from the instant in which the message is sent by the source and the instant in which the message is received at the sink. In particular, the adaptive scheme introduces an average latency significantly lower than the optimal fixed scheme (464 ms vs. 798 ms in the multi-hop

³ In the all-in-range scenario the average duty cycle for TAG is not 100% as one would expect since all nodes should be at 1 hop from the sink. This is because in some few cases nodes located at the upper corners of the grid (see Figure 2-a) associate with intermediate nodes thus producing a multi-hop topology in this scenario as well.

unbalanced scenario). In addition, the obtained latency is an order of magnitude lower than with TAG.

The delivery ratio is shown in Figure 3-c. All the sleep/wakeup schemes provide approximately the same delivery ratio (above 90%), and there is no significant difference when passing from one scenario to another. However, we can see that the adaptive algorithm is robust against the problems related to schedule exchange and maintenance due to beacon losses.

4.2 Impact of message size

In this section we investigate the impact of the message size – we have considered 20, 60 and 114 bytes in the message payload – on the performance of the different schemes. For the sake of space, we limit our analysis to the multi-hop unbalanced scenario (results are similar in the other scenarios as well).

Figure 3-d shows the average duty-cycle obtained by the different sleep/wakeup schemes. Clearly, for the adaptive and the fixed staggered schemes, it increases with the message size, while it remains approximately constant in TAG. It is worth noting that the adaptive scheme is able to automatically tune the length of the TI in this case as well, and this explains why its duty-cycle is lower than the other schemes.

For reasons similar to the ones presented in the previous subsection, also the latency (Figure 3-e) increases with the message size, even though the impact on the estimate is limited in this case. Again, the adaptive scheme has the lowest latency among the staggered approaches.

As in the previous section, the delivery ratio – with a given message size – is almost the same for the different schemes. As longer messages have a higher probability to be corrupted due to link errors and collisions, the delivery ratio clearly decreases when the message size increases. This result is useful to explain some of the findings of the next subsection.

4.3 Impact of data aggregation

Finally we consider the impact of aggregation on the performance of the different duty-cycling schemes. To this end we defined two types of aggregation.

- *Min/Max Aggregation.* Each parent node aggregates all received messages into a single message of the same size.
- *Compact Aggregation.* Each parent node merges all received messages into a single data block, which is eventually split across messages with the maximum payload size, i.e. 114 bytes (excluding overhead) in our case.

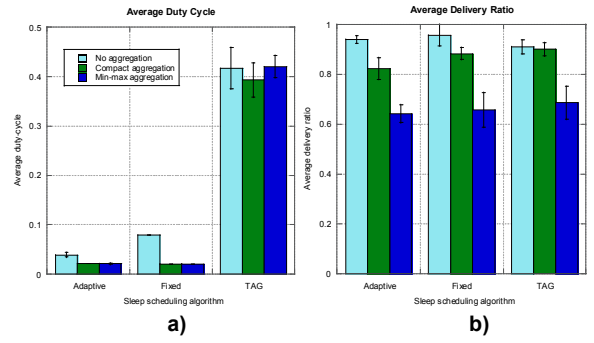


Figure 4. Performance indices for aggregation

We start considering the average duty-cycle, illustrated in Figure 3-a. We can see that aggregation, by reducing the amount of data to be transmitted, actually reduces the activity as well for the adaptive and the fixed staggered schemes, while the same is not true for TAG. In detail, the average duty-cycle is almost the same for the adaptive and the fixed staggered schemes. This happens because, due to aggregation, the adaptive algorithm uses the lowest TI value of one slot (150 ms) almost for each hop. In addition, we can see that the difference between the two aggregation schemes is not relevant in this case. This is due to the number of nodes, which is not so high, so that the impact of data aggregation is limited. If we consider the delivery ratio, instead, we can see the impact of a specific aggregation scheme is significant (Figure 4-b). Notice that, in this case, we assumed that if a message is correctly received by the sink, all data in the aggregate are assumed to be delivered successfully. If we focus on a given aggregation scheme, we can see that the different sleep scheduling approaches almost achieve the same value for the delivery ratio, substantially confirming the trend of Figure 3-c. Clearly, the more aggressive is the aggregation scheme (i.e., the more information is summarized in a message), the more relevant is the impact on the delivery ratio caused by message losses. This is because Min/Max aggregation provides a delivery ratio lower than the other schemes (the delivery ratio drops to values under 70%). As far as Compact aggregation, this scheme reduces the number of messages but increases their size. Therefore, the gain in terms of message loss probability due to aggregation is compensated by the negative effect caused by the increased message size (recall Figure 3-f).

5. Conclusions

In this paper we have presented a prototype implementation and experimental evaluation of the

ASLEEP protocol for power management in sensor networks for environmental monitoring applications. By adjusting dynamically the wakeup period of each single sensor node to its current traffic pattern, the proposed protocol reduces both the energy consumption and message latency. In addition, it is able to adapt to changes in the operating conditions. Through a set of experiments on a real testbed, we have shown that the proposed solution provides better performance, in terms of reduced energy consumption and message latency, in comparison with other similar staggered approaches. These experimental results confirm previous, very promising, simulation results obtained in more general scenarios, thus showing that the proposed solution is practical and suitable for real-life applications.

References

- [1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless Sensor Networks: a Survey", *Computer Networks*, Volume: 38, No: 4, March 2002
- [2] V. Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks", in *Proc. of Sensys '03*, Los Angeles, CA, Nov. 2003
- [3] G. Lu, B. Krishnamachari, and C.S. Raghavendra, "An Adaptive Energy-efficient and Low-latency Mac for Data Gathering in Wireless Sensor Networks", in *Proc. of PDSPP '04*, Pages: 224, April 2004
- [4] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks", in *Proc. of OSDI '02*, 2002
- [5] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring", in *Proc. of WSNA 02*, Pages: 88-97 Atlanta, GA, USA, 28 September 2002
- [6] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Sensor Networks", in *Proc. of SenSys '04*, November, 2004
- [7] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy Aware Wireless Microsensor Networks", *IEEE Signal Proc. Mag.*, 19(2), March 2002
- [8] G. Tolle et al., "A Macroscopic in the Redwoods", in *Proc. of SenSys 05*, San Diego, 2-4 November 2005
- [9] W. Ye, J. Heidemann, and D. Estrin, "An Energy-efficient MAC Protocol for Wireless Sensor Networks", in *Proc. of IEEE Infocom '02*, New York, June 2002
- [10] G. Anastasi, M. Conti, M. Di Francesco, A. Passarella, "An Adaptive and Low-latency Power Management Protocol for Wireless Sensor Networks", in *Proc. of MobiWac 2006*, Torremolinos (Spain), October 2, 2006
- [11] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "How to Prolong the Lifetime of Wireless Sensor Networks", *Chapter 6 in Mobile Ad Hoc and Pervasive Communications* (M. Denko and L. Yang, Editors), American Scientific Publishers, to appear.

- [12] Tmote Sky Platform, MoteIV Corporation, <http://www.moteiv.com/products/tmotesky.php>
- [13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors", *SIGPLAN Not.* 35, 11, pp. 93-104, Nov. 2000
- [14] S. Ping, "Delay Measurement Time Synchronization for Wireless Sensor Networks", *IRB-TR-03-013*, Intel Research Berkeley Lab, 2003
- [15] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks", *Proc. of SenSys 03*, Pages: 14-27, Los Angeles, CA, November 2003
- [16] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks", *IEEE/ACM Trans. Netw.* n.12(3), Jun. 2004
- [17] B. Hohlt and E. Brewer, "Network Power Scheduling for TinyOS Applications", in *Proc. of DCOSS 2006*, Pages: 443-462, San Francisco, CA, June 2006

APPENDIX A: Talk Interval Estimation

To estimate the duration of the talk interval (TI) in the next communication period (CP), a parent node measures the following quantities.

- *Message inter-reception time* (Δ). The message inter-reception time is the difference between the time instants at which two subsequent messages are correctly received.
- *Number of received messages* (n_{pkt}). The total number of messages received in a single CP.

The time expected to get all messages sent by the children in the next communication period is then estimated as $TI_{est}^{m+1} = \bar{\Delta} \cdot n_{pkt}^{max}$, where $\bar{\Delta}$ and n_{pkt}^{max} are the average inter-reception time and the maximum number of received messages over the last L communication periods, respectively. Using n_{pkt}^{max} is a conservative choice to cope with lost messages. The above estimate is then increased to also include the beacon period. Finally, the expected talk interval is rounded to an integer number of time slots, whose duration is denoted by q . As a result of the operations described above, a value TI_{rnd}^{m+1} is obtained. However, advertising TI_{rnd}^{m+1} to children as the next talk interval might lead to some flapping of the protocol parameters. To smooth the variation of the TI estimate, we also define two guard bands g_1 and g_2 . The next TI to advertise (TI^{m+1}) is then computed as follows:

$$\begin{aligned} & \text{if } (TI_{rnd}^{m+1} - TI^m \geq g_1) \text{ then } TI^{m+1} = TI_{rnd}^{m+1} \\ & \text{else if } (TI^m - TI_{rnd}^{m+1} \geq g_2) \text{ then } TI^{m+1} = TI^m - q \end{aligned}$$

The above rules show that increases in the talk interval are managed by the algorithm less conservatively than decreases, in order to minimize the probability that a node misses messages from its children.