T-AAD: Lightweight Traffic Auto-ADaptations for Low-power MAC Protocols

Georgios Z. Papadopoulos[†], Julien Beaudaux[*], Antoine Gallais[†], Thomas Noël[†]
[†]ICube Laboratory, University of Strasbourg, France
[gpapadopoulos, gallais, noel]@unistra.fr
[*]Norwegian University of Science and Technology (NTNU), Trondheim, Norway
julien.beaudaux@item.ntnu.no

***Abstract*—After many successful deployments, Wireless Sensor Networks (WSNs) now appear open for a paradigm shift in traffic resulting from applications. Consequently to the traditional event or time-driven a priori known traffic patterns, those networks face occasional, bursty and unanticipated multi-hop data transmissions. This specific case is rarely addressed by preamble-sampling Medium Access Control (MAC) protocols. Such homogeneously configured solutions avoid network disconnections and isolated nodes, yet lead to long periods of channel occupancy, increased delays and energy-consumption. We here present T-AAD, a Traffic Auto-ADaptive mechanism, specifically designed to address the previously reported phenomenon through the introduction of heterogeneous node configurations in the network. T-AAD dynamically and locally adapts the MAC configuration depending on the actual and expected traffic load, without endangering network connectivity nor overall network performances. T-AAD is therefore compliant with Low Power Listening (LPL) based MAC protocols. In this paper, we evaluate our proposed mechanism and we demonstrate that it provides significant gains in terms of delay and energy consumption, when compared to respective research work available in the literature (e.g. MaxMAC, AADCC).**

***Index Terms*—Wireless Sensor Networks, Medium Access Control, Energy Efficiency, Runtime Traffic Auto-adaptivity, Localized Mechanism**

I. Introduction

The previous decade saw the emergence of Wireless Sensor Networks (WSNs) deployments, with applications ranging from wildlife monitoring to clinical medical and home-care. In such deployment, energy-efficiency is one of the most important parameters, since nodes have to save energy to meet the lifetime requirements of typical applications. Even more, WSNs have come to maturity, thus allowing more complex applications and traffic patterns. This state can occur in event-driven traffic models, where nodes only send their readings upon detection of a specific event determined by the application, and when nodes may store their readings and send them in a row sometime after (e.g. when the channel is occupied, when the sink becomes unreachable). This situation implies increased channel occupancy for limited periods of time, at the cost of deteriorating the average network performance.

The Medium Access Control (MAC) layer manages the communications between nodes. In WSN, it is also responsible for switching the radio device ON and OFF at regular intervals. This duty-cycling functionality results in a fundamental trade-off between energy consumption and network performance (i.e. latency). Numerous MAC protocols have been devised [1], but to the best of our knowledge very few of them address the needs implied by the presence of variable traffic in the network (e.g. MaxMAC, AADCC). As a result, no successful WSN deployment with auto-adaptation to the traffic has been experienced.

In this paper, we consider contention-based MAC protocols [1] mainly because they are scalable (e.g. local decisions) and robust to network changes (e.g. mobility). In these protocols, nodes in the network asynchronously check the radio channel for incoming packets at regular intervals. In between, they turn OFF their radio to save energy. The period between two channel samplings is referred to as the Sleeping Time (ST). Then, a node expecting to transmit will have to send beforehand a preamble longer than the ST, to ensure that its target is awakened. While a longer ST allows nodes to save energy, it reduces the amount of traffic they can handle. Conversely, a shorter ST allows nodes to handle more traffic, at the cost of increased energy consumption.

Since two nodes with different MAC configurations may be unable to communicate, most of the WSN deployments consider homogeneous (i.e. identical for all nodes) and static (i.e. invariant over the time) [2], [3]. But avoiding any partition of the network is a situation far from ideal, since performance and energy gain could be increased by providing a dedicated configuration to each node, according to its current traffic load.

In this paper, we introduce T-AAD, a Traffic Auto-adaptive mechanism. T-AAD anticipates traffic load variations by tuning the MAC parameters according to the upcoming traffic volume of the concerned nodes. Using T-AAD, a node decides autonomously from its own parameter setting, thereby trying both to increase the network lifetime (i.e. reduce to minimum the energy consumption) and to decrease the latency.

The contributions presented in this paper are threefold.

- We first introduce T-AAD, an algorithm that automatically adapts the MAC parameters at runtime. Assuming many-to-one data traffic (sink-rooted routing), T-AAD is compliant with preamble-sampling MAC protocols.
- We then demonstrate through a mathematical study of its behavior that it quickly adapts to traffic variations. Thus, we show to what extent it allows for reduced energy consumption at both the receiver and sender sides, along with delay and channel occupancy reductions, when compared to ContikiMAC [4] and MaxMAC [5].

- Finally, we perform a thorough performance evaluation, both through simulation (i.e. Cooja simulator [6]) and experimental study over FIT IoT-LAB [7] testbed. In addition, we compare our mechanism both with a statically configured network using X-MAC [8] and a state-of-the-art auto-adaptive mechanism such as AADCC [9].

The rest of this paper is organized as follows. In Section II, we discuss related work in auto-adaptation mechanisms. Section III provides the detailed design and description of the methodology that T-AAD follows. Section IV, presents the both simulation and experimental performance evaluation results of our proposal. Finally, Section V provides conclusive remarks and future directions.

II. Background and Related Work

During the previous decade, many contention-based MAC protocols were proposed. Two are the most prominent categories: Low-Power Listening (LPL) and Low-Power Probing (LPP). In LPL-based protocols, nodes sleep most of the time and wake-up periodically (i.e. asynchronously) to sample the channel. If a node detects a carrier, it keeps its radio ON to receive the associated message; otherwise it returns to sleep. The transmitter sends preambles for a period longer than the ST so that the receiver can detect the carrier. In LPP-based protocols the opposite happens: the sender first waits for a probe from the intended receiver and then it sends the packet. In this study we focus on the LPL family of protocols, thus, below we review the LPL-based key contributions from the literature that are related to auto-adaptation. However, the principles that govern our mechanism can be swiftly transposed to LPP protocols as well.

B-MAC [10] was among the first of the LPL protocols. Nodes can have independent schedule for an awake and for a sleep period, and send data packets with very long preambles to ensure that the intended receiver will stay on upon sampling the medium. Post B-MAC protocols such as X-MAC [8], MX-MAC [11], SpeckMAC [12] or ContikiMAC [4] replace the long preamble used by B-MAC with series either of strobes (e.g. X-MAC) or of data packets (e.g. ContikiMAC). Hence, the transmitter repeatedly sends strobes or data that contain the address of the receiver. On the other side, the intended receiver replies with an ACK and stays ON until the data transmission is complete. Nodes that have a different address from the one that is indicated in the strobe go back to sleep. ContikiMAC, in addition addresses the false wake-up problem with phase lock and fast sleep techniques which make the transceiver turn OFF the radio for longer period of time. Moreover, with BoX-MACs [13] or ContikiMAC, a node can be configured so that to keep the radio ON for short time, right after receiving a packet in order to cope with consecutive packets. These protocols greatly reduce overhearing on the nodes and achieve low power operation at both the receiver and transmitter sides. However, they do not meet our goals of automatic and on-the-fly localized adaptation of duty-cycle configurations in the network, since all the nodes of the network operate homogeneously. In [11], Merlin *et al.* present MiX-MAC where nodes switch schedules from a pool of MAC protocols (i.e. X-MAC, MX-MAC and SpeckMAC) based on some parameters (e.g. packet size). However, since it switches between post B-MAC protocols, it has the same disadvantages.

pTunes [14] dynamically adapts the networks MAC configuration depending on the traffic load. Along with every message sent to the sink, information about the networks status (e.g. traffic load, loss-rate, delays) is added, using piggybacking. Over the successive packet receptions, the sink updates network statistics and computes a MAC configuration corresponding to the actual state of the network. Then, it broadcasts this value to the whole network, so that each node can fit to it. This approach provides gains in terms of performance (e.g. delays, loss-rate) as the network will dynamically adapt to the traffic load. But, it requires a broadcast mechanism relying on node synchronization. Also, this scheme adapts the nodes MAC configuration globally, providing to each of these the same MAC values. Then, the nodes being configured homogeneously over the network, pTunes could fail to address local traffic load changes.

In MaxMAC [5], each node keeps estimating the rate of incoming packets. Nodes change their duty-cycle by allocating so-called Extra Wake-Ups when the rate of incoming packets reaches predefined threshold values, and de-allocate them when the rate drops below the threshold again. Hence, during the first two thresholds nodes double their duty-cycle while in the last (third) threshold, the sensor nodes start operating in a CSMA fashion without going to sleep. The receiving nodes use ACK packets to inform about the change in the duty-cycle and the duration of it. Moreover, switching in the CSMA mode, the receiver node will resulting a higher energy consumption at its side, as we demonstrate in Section III.

In [9], the authors present AADCC (Asymmetric Additive Duty Cycle Control) that is based on the number of consecutive packet transmissions. Hence, each node increments its sleep time by $100\ ms$ when five consecutive packets are transmitted successfully to the destination while for each failed packet the sleeping time is decreased by $250\ ms$. Jurdak et al. [15] propose Adaptive Low-Power-Listening (ALPL) a cross-layer framework for network-wide power consumption optimization and load balancing in WSN through greedy local decisions. The algorithm periodically gathers neighborhood state information, performs local calculations on the gathered state, and modifies the local configuration of routing and MAC layer accordingly. ZeroCal [16], is an asynchronous scheduling approach that configures the MAC parameters (i.e. wake-up intervals) between parent and child nodes on the fly to minimize and to balance the energy consumption. To compute the wake-up intervals, an energy consumption model is used based on collecting statistics (e.g., packet loss, network topology) on each node and all its children, and is made at fixed intervals or when the number of sent packets of a child exceeds a given threshold. All AADCC, ALPL and ZeroCal algorithms are dedicated to extend the network lifetime. This leads to some nodes using shorter intervals than any of its children in order to prevent preamble misses. Hence, they manage to rapidly adjust duty-cycles on traffic increase, but adapt slowly to reductions in load. Moreover, these proposals may not guarantee the end-to-end delay bounds as the wake-

MAC Protocol	Traffic Dependent	Decentralized decisions	Heterogeneous configurations
{B ,[10] X [8], MX [11], Speck [12], MiX [11]}-MAC	☐	☑	☐
pTunes [14]	☑	☐	☐
{Contiki [4], BoX [13]}-MAC	☑	☑	☐
MaxMAC [5], AADCC [9], ALPL [15], ZeroCal [16]	☑	☑	☑
T-AAD proposal	☑	☑	☑

TABLE I
SUMMARY OF STATE-OF-THE-ART CONTRIBUTIONS : TRAFFIC DEPENDENCY, REQUIRED KNOWLEDGE FOR DECISION MAKING AND POSSIBILITIES OF HETEROGENEOUS CONFIGURATIONS WITHOUT ENDANGERING NETWORK CONNECTIVITY

up interval can be extended to save nodal energy. Finally, it takes time and induces communication overhead to adapt, as it requires several messages in order to perform some calculations first and then to change the configuration. This may lead to excessive latency when facing bursty traffic.

These solutions may not satisfy our goals of addressing dynamic and bursty traffic, in a highly reactive and low energy consumption manner.

Hence, as summarized in Table I, the static protocols (i.e. preconfigured static MAC values, such as {B, X, MX, Speck, MiX}-MAC solutions) do not provide any on-the-fly auto-adaptation and maintain homogeneous MAC configurations for the whole set of deployed nodes. On the other hand a protocol like pTunes, although dynamically adapting to the traffic load, relies on a centralized decision making system that induces broadcasted control messages from the sink node to all nodes. Moreover, it also maintains homogenous LPL configurations, thus failing to cope with non uniform traffic distributions in an energy-efficient manner (e.g. burst traffic upon event occurrence). Regarding ContikiMAC, dynamic and bursty traffic can be handled. But, this proposal relies only on waiting periods after each packet reception. Therefore, no adaptation on the sender is allowed and the same preamble and sampling periods are kept throughout the deployment, thus relying on homogeneous configurations that prevent any further energy savings.

Finally, solutions such as MaxMAC or AADCC appear as the most relevant in our targeted context, being both adaptive and traffic-aware. We therefore selected these solutions as best candidates for further comparisons during our evaluation campaign, as detailed throughout this paper.

III. T-AAD DESIGN

In this paper we propose T-AAD, a lightweight traffic auto-adaptive algorithm for WSNs facing variable and dynamic traffic loads. The idea behind T-AAD is to automatically adapt MAC parameters, in reaction to traffic load variations. In particular, this mechanism tunes the MAC parameters based on the amount of packets that a node expects to transmit.

We propose a receiving node to switch from long to short ST mode for the period of traffic variation, and then back to the long mode. Thus, the node adopts a short ST configuration for a certain time in order to quickly handle a high traffic load, and then it switches back to long ST during periods of sporadic traffic, thus maintaining a low-power MAC configuration. Figure 1 illustrates X-MAC protocol, without and with the

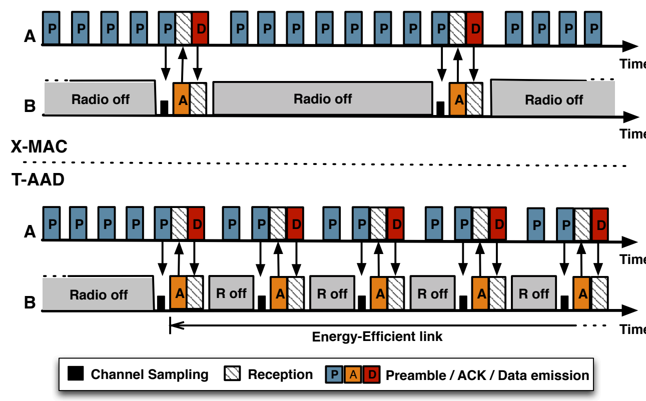


Fig. 1. Improvement introduced by the T-AAD mechanism over X-MAC

T-AAD mechanism. Two nodes (A and B) are to handle a bursty traffic. Node A sends to B and should be allowed to reduce the preamble period. Node B is receiving from A and should sample the medium more often in order to cope with incoming bursty traffic. This way, both the A and B nodes will be able to handle the traffic variation quickly (i.e. delay decrease and throughput increase) and in an energy efficient fashion (i.e. extending the battery lifetime).

In this section, we detail how A and B can agree on both preamble and channel sampling periods, thus experiencing lower energy consumption (especially for A since it has a shorter preamble duration) and increasing reactivity thanks to shorter delays (especially for B that handles more received packets in the same time frame). We need to emphasize the fact that no centralized computation (i.e. no communication overhead for control traffic) is required while no connectivity loss can be experienced.

A. Detailed description

Our proposal is based on a mechanism where a receiver node smoothly adjusts to the traffic demands for the time that the traffic load variation takes place. In order to achieve it, one piece of information is included in each packet. This allows the indication of the number of packets (i.e. through MAC or application queue) that a node expects to send in a row (e.g. queue storage). This value is used to calculate the time (here referred as T_{adapt}) which a node stays in the short ST mode.

All nodes start with identical long ST configurations, referred to as ST_{max} (e.g. 500 ms, 1000 ms), depending on the initial user deployment configuration to face sporadic traffic.

Algorithm 1: Functionality of the T-AAD mechanism

begin
  when *reception of packet P* **then**
    if ST_{max} **then**
      if $P.Q_{len} > 1$ **then**
        $calculate\ T_{adapt}$;
        $set_{ST}(ST_{min})$ for T_{adapt};
      else
        $continue$ with (ST_{max});
      end
    else
      if $P.Q_{len} > 1$ **then**
        $calculate\ new\ T_{adapt}$;
        if $new\ T_{adapt} > current\ T_{adapt}$ **then**
          $set_{ST}(ST_{min})$ for $new\ T_{adapt}$;
        else
          $continue$ with $current\ T_{adapt}$;
        end
      else
        $continue$ with $current\ T_{adapt}$;
      end
    end
  end
end

Once a receiving node detects a traffic load variation, it will calculate the T_{adapt} according to the information at the first received packet.

Then, it will switch its ST mode to minimum, depending on the hardware, referred as ST_{min} (e.g. 32 ms, 64 ms), and will return to ST_{max} as soon as the traffic burst is expected to end, in order to cope with the nature of the transmission. Hence, a node with the T-AAD mechanism running on top of any static MAC protocol gains energy when using a long ST mode, and performs better in delay when switching to short ST. Thus, it is expected to profit from both situations. The detailed function of T-AAD is presented in Algorithm 1. Note that one node implementing this Algorithm regularly updates a timer according to T_{adapt} period. Upon the expiration of this timer, the node switches back to a long ST mode.

B. Theoretical analysis

T_{adapt} designates the actual time that a node will stay in the short ST mode. Let ST_{max} be the time period for long ST and ST_{min} for short ST respectively. Q_{len} indicates the total number of packets that a transmitter is expected to send, and finally let M_{err} be the margin of error. We consider M_{err} to cope with the potential packet retransmissions. This situation can be caused by collisions or interferences in the network. Hence, in order to ensure that the calculated period is at least as long as the traffic load change (including MAC retransmissions), a margin of error is added depending on deployment conditions (e.g. loss-rate, perturbations). Hence, this observation can be modeled and computed as follows:

$$T_{adapt} = ST_{max} + (Q_{len} - 2) \times ST_{min} \times (1 + M_{err}) \quad (1)$$

Let us assume that a node A has n packets to be transmitted towards the node B. Thus, the queue length (i.e. Q_{len}) is equal to n. There are $n-1$ sleeping periods among the n consecutive packets. The receiver is in the ST_{max} mode during the first period, and the rest $n-2$ switches in short ST (i.e. $n-2$ times the ST_{min}), thus inducing highly reactive changes in the MAC configuration. As a result, T-AAD is able to handle varying traffic efficiently. Finally, since each packet contains Q_{len}, node B first will calculate the T_{adapt} before switching to ST_{min} for T_{adapt} period.

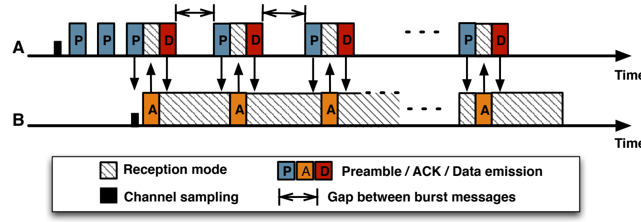


Fig. 2. CSMA-based solutions

C. Advantage of T-AAD over CSMA-based solutions

In existing protocols such as BoXMACs [13] or Contiki-MAC [4], each node can be configured in order to keep its radio ON for a while, right after receiving each packet. As a result, they turn into the CSMA mode during traffic load increase. In MaxMAC [5], once the incoming packet rate reaches the last threshold, the nodes switch in a CSMA fashion as well (see Figure 2, the CSMA version of MaxMAC).

It is worth pointing out the reason why we aim to reduce the channel sampling frequency to minimum rather than eliminating it completely and switching into the basic CSMA mode and then recovering to the preamble-sampling mode again. In particular, we observe in both simulation and experimentation campaigns that on the transmitter side there is a time interval between two consecutive packet transmissions because of the necessity to have a Clear Channel Assessment (CCA) before each message transmission). Note that, one of the most important parameters of the radio component is the CCA threshold, which indicates if the radio channel is free or busy. In fact, while the CCA is enabled, packets will be transmitted only if the channel is free. Hence, keeping the radio active at the receiver side during this period has a significant impact on the energy performances. Knowing that most of the power consumption is spent during the reception or transmission, and even more after our mathematical analysis, we notice that switching the radio between each mode (i.e. ON and OFF) provides more energy gain than keeping it ON. Below is the formula which provides us with the energy gain that we obtain compared with CSMA-based solutions.

$$E_{save} = (P_{rx} - P_{off}) \times (X - 1) \times (T_{int} - T_{Soff} - T_{Son}) \quad (2)$$

This equation is always true if and only if the following three concepts are fulfilled:

- The consumed reception power (i.e. P_{rx}) must be greater than the power consumed during the idle mode (i.e. P_{off}). According to most current hardware characteristics, this statement holds.

- X, represents the number of intervals between n consecutive packets. During high traffic loads, there are multiple packet transmissions, thus, X is greater than 1.
- Finally, the time between two consecutive packets (i.e. T_{int}) must be greater than the sum of times the radio switches ON and OFF (i.e. T_{Soff} and T_{Son}). This value can be retrieved from the radio component datasheet or determined after initial simulation and experimental campaigns. Indeed, during our evaluation, we experienced intervals ranging from 20 to 26 ms.

IV. Performance Evaluation

The mechanism presented in this paper can be applied to various preamble-sampling protocols (e.g. X-MAC, Speck-MAC, ContikiMAC). To evaluate our mechanism we picked the X-MAC [8] protocol since many real WSN and Internet of Things (IoT) deployments and recent scientific contributions rely on this protocol [14], [17]. Moreover, it was already pointed out that X-MAC is a serious alternative to the MAC layer defined in the 802.15.4 standard [2]. Hence, we believe that X-MAC is a generic protocol above which we show that our proposal can work. X-MAC optimizes the LPL mechanism by allowing intended receivers to reply to strobed preamble with an ACK, which triggers a data transmission. Thus, it stays ON until the data transmission is complete. This scheme mitigates both overhearing on non-recipient nodes and long preamble transmissions on senders.

T-AAD aims at managing dynamic traffic at best, by adapting MAC parameters locally and at runtime. In the previous section, we exposed the design of this mechanism, and detailed to what extent it may reduce energy consumption and delay when compared to major contributions from the literature. In this section, we present a thorough performance evaluation, both through the Cooja simulator [6] with Sky motes and experimental study over FIT IoT-LAB, a very large scale WSN testbed [7]. In addition, we implemented and compared our mechanism both with a statically configured network using X-MAC [8] and a state-of-the-art auto-adaptive mechanism such as AADCC [9] on top of the Contiki OS [18], in order to have a fair and thorough comparative study.

A. Simulation setup and evaluation

1) Simulation setup: In this study, we assigned 500 ms for ST_{max} and 32 ms for ST_{min} respectively as ST configurations for T-AAD and AADCC both for simulation and experimentation. On the other hand, for pre-configured X-MAC, we kept the standard format with ST of 32, 125, 250 and 500 ms (i.e. X-32, X-125, X-250 and X-500 respectively). The maximum number of retransmissions was 3 for all protocols. At the routing layer, we rely on a broadly used scalable gradient protocol which generates a routing tree rooted at the sink (i.e. using a number of hops as a metric). Note that the induced control message overhead is limited since a single message originates from the sink (including a rank equal to 0) [19]. All receiving nodes (i.e. in the communication area of the sink station) update their rank before forwarding the message. The process is kept running while new control messages are initiated only upon reception of a better rank information.

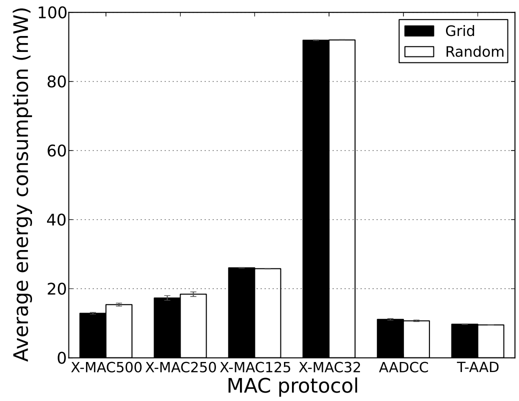


Fig. 3. Average energy consumption

This routing protocol was chosen for its performances under realistic conditions [3].

Our simulation scenario involves 50 wireless sensors that are either randomly or uniformly distributed (i.e. grid) in an area of 50×50 meters. The nodes use a time-driven application in which 10 packets are sent in a row, every 500 seconds, with a random initial backoff to avoid synchronous traffic bursts. We chose a 10 byte data size for both the simulation and the experiment, which corresponds to the general information used by monitoring applications (e.g. node ID, packet sequence, sensed value). For the sake of clarity and easier initial analysis, before running our tests over our real sensor network testbed, we used a radio model based on disks where each wireless sensor has a -10 dBm power transmission, thus imposing multi-hop communications to reach the sink station (i.e. up to four hops). The simulation lasts for 35 minutes.

The results hereinafter present the gain of our proposal in terms of energy consumption and delay, compared to a state-of-the-art auto-adaptive protocol AADCC and to a homogeneously pre-configured X-MAC. More importantly they show that heterogeneous configurations can be determined independently at each node in a localized fashion, without risking any partitioning of the network. The details of the simulation setup are exposed below in Table II.

2) Simulation evaluation: We now present the simulation results and the comparison with AADCC.

a) Energy consumption: All energy results were obtained by using the Energest module of Contiki [20]. This module monitors in real-time the radio and CPU usage by saving the duration radio or CPU spent in each state (awaken, sleeping, receiving, transmitting data, etc.). This information is then combined with the component datasheets energy value for each state, in order to provide an accurate calculation of the node energy consumption.

In Figure 3, the average energy consumption per second for the whole network is presented for both grid and random topologies. T-AAD and AADCC consume less energy network-wide than any of the pre-configured X-MAC. In particular T-AAD reduces the energy depletion by about 37% when compared to X-125. The results show that the preamble length has a straightforward impact on energy dissemination.

Platform parameters	Cooja Simulator Value	IoT-LAB testbed Value
Topology	Grid / Random ($50m \times 50m$)	Regular grid ($10m \times 8m \times 3m$)
Number of nodes	50 fixed sensors	80
Number of sources	49	79
Node spacing	7.14 $meters$	1 $meter$
Simulation & Experimental parameters	**Value**	**Value**
Duration	35 $minutes$ / 2100 $seconds$	70 $minutes$ / 4200 $seconds$
Application model	Burst 10 $pkts/500\ s$	Burst 10 $pkts/1000\ s$
Number of events	2000	2400
Payload size	10 $Bytes$ (+6 $Bytes$ MAC header)	
Routing model	Gradient [19]	
MAC model	T-AAD: ST_{min}=32 ms, ST_{max}=500 ms AADCC [9]: ST_{min}=32 ms, ST_{max}=500 ms X-MAC [8]: $ST = 32$, 125, 250 or 500 ms	
Maximum retries	3	
Margin of error (M_{err})	15% (see Equation 1)	
Hardware parameters	**Value**	**Value**
Antenna model	Omnidirectional CC2420	Omnidirectional CC1101
Radio propagation	2.4 GHz	868 MHz
Modulation model	O-QPSK	GFSK
Transmission power	$-10\ dBm$	$-20\ dBm$

TABLE II
SIMULATION & EXPERIMENTAL SETUP

In case of low traffic, the total energy consumption is higher for short preamble lengths due to more periodic channel samplings. Conversely, in the case of heavy traffic loads, long preambles consume more due to the constant presence of transmission and reception of the preambles. As an example, the energy dissemination is higher for nodes that forward more (i.e. nodes located around the sink). Globally during the conducted experiments we observed that T-AAD performs slightly better than AADCC. This is mainly due to the long preambles that AADCC necessarily induces for each adaptation process.

b) One-hop delay: Figure 4 depicts the average one-hop delay per packet for all nodes. The one-hop delay includes the initial back-off, the channel sampling period, potential congestion back-offs, potential retransmission delay and the preamble length. With an average one-hop delay of 108.08 ms, T-AAD displays well performance. Even though T-AAD and AADCC have similar energy consumption, in terms of delay T-AAD performs almost five times better than AADCC. Thus, according to these results, T-AAD anticipates the traffic load variations better. Overall, all LPL configurations perform identically or worse in random topologies. This is mainly due to the potential bottleneck links that are more prone to appear in random topologies, having as a result nodes to handle heavy traffic. X-500 has the worst performance (i.e. 1688.6 ms) due to the very high channel occupancy (i.e. high congestion) the radio medium is almost never available to transmit a packet. This reveals that there is a high probability of provoking congestion back-offs in the network, especially for the nodes that are located one-hop away from the sink, having as a result the increase of the one-hop delay around the sink. Furthermore, these higher delays are also due to MAC retransmissions. Indeed, due to radio channel competition and hidden terminals, larger proportions of sent packets require several retransmissions before being acknowledged.

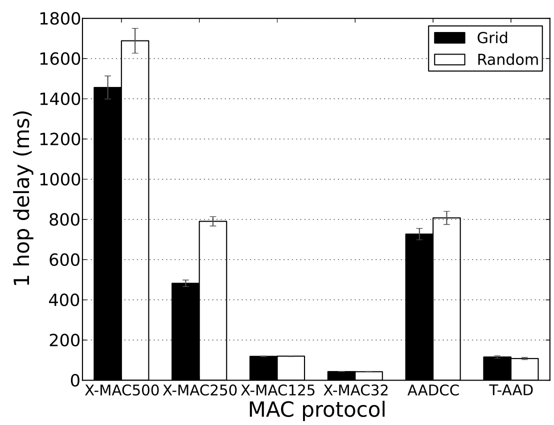


Fig. 4. Average one-hop delay

B. Experimental setup and evaluation

1) Experimental setup: Setting up a complete WSN deployment is a very complex task [3]. To further evaluate the performance of the T-AAD mechanism, we performed a number of experiments. Our thorough empirical analysis was conducted over the Strasbourg platform of the open large scale FIT IoT-LAB [7] testbed. This platform is structured as a 3D grid of 240 fixed nodes. Each node is equipped with a Texas Instruments CC1101 radio chipset and a MSP430 microcontroller. In order to limit border effects due to excessive radio reflections, we decided to work with 80 nodes of the same 2D layer. Still, the radio environment remained highly realistic, especially as the remaining nodes were free to be used by other experiments running at the same time over the testbed (e.g. potential radio interferences). All 79 nodes are randomly selected as data sources. The nodes implement a time-driven application model, and each of them transmits 10 packets in a row every 1000 seconds with an emission power at $-20\ dBm$, thus imposing multi-hop transmissions to reach the sink station (i.e. up to four hops). As per Equation 1, the margin of error was fixed at a realistic value of 15% of the total T_{adapt}. The sink is located at the top left of the area. The experiments last for 70 minutes.

2) Experimental evaluation: At first, we evaluated the general performances of the network, either configured homogeneously using X-MAC or implementing auto-adaptation mechanisms (AADCC and T-AAD). Figure 5 illustrates the network performances in terms of delay (i.e. average time for ten packets transmissions) and energy consumption, for each case of study. Both results outline the same aspects as for simulation. Indeed, we can observe that the delay increases respectively with ST, while energy consumption decreases, thus demonstrating that there is a strong antagonism between energy consumption and delays when considering homogeneously configured networks. Conversely, T-AAD allows the network to gain both in delay and energy consumption at the same time. In fact, T-AAD allows for delays similar to X-125 with an energy consumption lower than that obtained with AADCC or X-500. This phenomenon is due to both a better handling of messages during periods of high traffic load, and to lower sampling frequencies during calm periods in the network. Moreover, it is achieved in a localized fashion, without endangering network connectivity.

We also performed a mapping of the energy depletion throughout the network, in order to identify its repartition and disparity among the nodes. To do so, we measured the total energy consumption of each node for the whole duration of the experiments. It can be noted that it does not impact the protocol as energy monitoring is a service provided by the testbed through independent monitoring channels. The results of this evaluation are displayed in Figure 6 for T-AAD, AADCC and X-125. As can be observed, the energy consumptions of X-MAC and AADCC vary depending on the considered node. Some nodes, located at the extremities of the routing topologies consume only a little energy (about 8 W), while most nodes have to manage the traffic load and thus consume a greater amount of energy during the experimentation (130 W in average). On the contrary, with T-AAD, the energy consumption remains similar throughout the network (with the exception of the sink). Indeed, the nodes consume about 25 W, independent of the position of the node in the routing topology.

The previously presented phenomenon is due to the fact that the traffic load bears a strong influence on energy consumption (long preamble transmissions). As a consequence, nodes which have to relay information from their neighbors will have to remain active for a larger proportion of time than the nodes that are activated, only to send their own data. T-AAD, allows for a quicker management of packets, and thus nodes can go back to sleep after a short period only, which explains the homogeneity of its energy depletion. Hence, T-AAD appears to be more stable and less dependent on the traffic load, as well as providing significant energy consumption reduction when compared to any network homogeneously configured with X-MAC or with an auto-adaptive AADCC mechanism. In addition, the homogeneity of this energy depletion allows T-AAD to improve the network lifetime.

C. Summary of evaluation

In this section, we evaluated to what extent our T-AAD mechanism can enhance the management of versatile traffic by auto-adapting dynamically and locally the MAC parameters. We demonstrated both through simulations and experimental campaigns that our solution allows for energy savings, and delay decrease as well as channel occupancy reduction, when compared to any homogeneously pre-configured X-MAC version. Moreover, T-AAD outperforms state-of-the-art auto-adaptive solutions such as AADCC, with a 12% reduction in energy wastage along with a 58% decrease in terms of delays.

V. Conclusion and Future Work

In this study, we could show that addressing a scenario with traffic load variations in homogenous preamble-sampling configurations reveals a trade-off. On the one hand when configured with ST of 500 ms less energy is consumed but poor network performance is induced (e.g. high latency) and vice versa with ST 125 ms. Hence, after a thorough study of the state of the art, we proposed a novel approach to address this problem. T-AAD is an auto-adaptive algorithm that is compliant with LPL based MAC protocols. Our mechanism is based on local decisions at each node, made on the available information from data transmissions (e.g. acknowledgments, preambles), thus limiting overall communication overhead and being well suited for constrained devices and networks. T-AAD also exhibits the feasibility of being able to manage the heterogenous MAC configuration in a localized fashion, without endangering network connectivity and consequently to handle dynamic traffic (e.g. energy-efficiency, delay)

In this paper, we have performed an exhaustive performance evaluation both through the Cooja simulator [6] and an experimental study with the FIT IoT-LAB [7] testbed in order to accurately analyze the performance of our proposal. We compared our mechanism both with a statically configured network using X-MAC [8] and an auto-adaptive mechanism such as AADCC [9]. Our results show that T-AAD outperforms any preconfigured X-MAC setup since it automatically reaches the best MAC configuration for the time of the traffic load change. We also avoid transition phases and thus useless long preambles such as in AADCC, which allows T-AAD to outperform this solution. Therefore, T-AAD succeeds in addressing the energy/latency trade-off. Our ongoing work consists of further exploring this lead in mobile sensor and actor networks, and in particular by investigating the automated learning of traffic patterns.

References

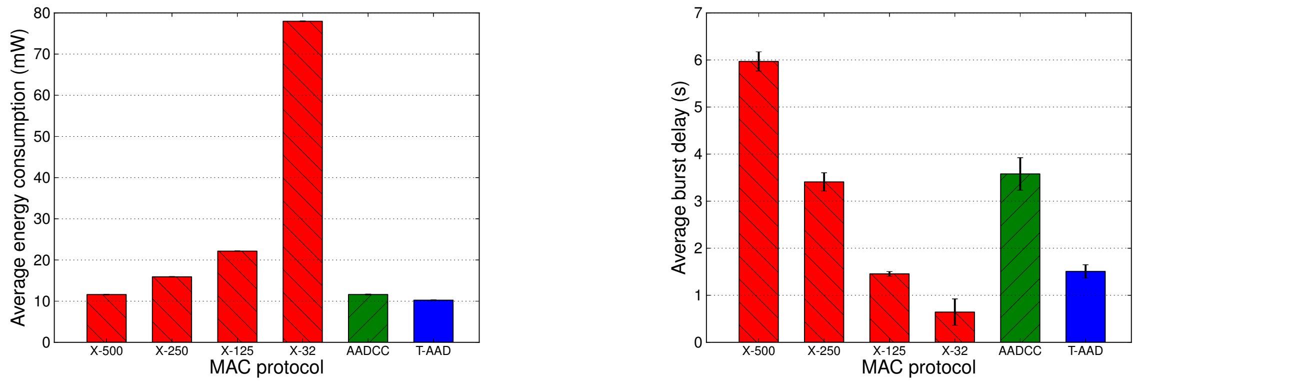
[1] C. Cano, B. Bellalta, A. Sfairopoulou, and M. Oliver, “Low Energy Operation in WSNs: A Survey of Preamble Sampling MAC Protocols,” in *Elsevier Computer Networks*, vol. 55, no. 15, pp. 3351–63, Oct. 2011.

[2] P. Suarez, C.-G. Renmarker, A. Dunkels, and T. Voigt, “Increasing ZigBee Network Lifetime with X-MAC,” in *Proc. of the ACM Workshop on Real-world Wireless Sensor Networks (REALWSN)*, 2008, pp. 26–30.

[3] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, “SensorScope: Out-of-the-Box Environmental Monitoring,” in *Proc. of the ACM/IEEE International Conf. on Information Processing in Sensor Networks (IPSN)*, 2008, pp. 332–343.

[4] A. Dunkels, “The ContikiMAC Radio Duty Cycling Protocol,” *Swedish Institute of Computer Science Technical report T2011:13*, 2011.

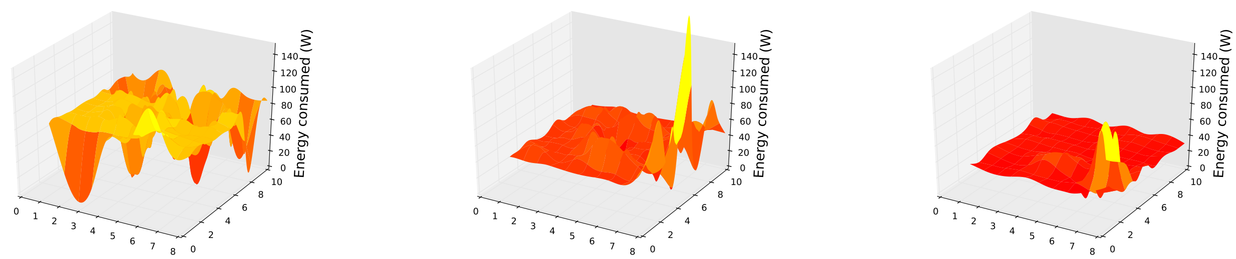
[5] P. Hurni and T. Braun, “MaxMAC: a Maximally Traffic-Adaptive MAC Protocol for Wireless Sensor Networks,” in *Proc. of the 7th European Conf. on Wireless Sensor Networks (EWSN)*, 2010, pp. 289–305.



(a) Average energy consumption

(b) Average burst delay

Fig. 5. Performances of MAC protocols in a complete network setup.



(a) X-125

(b) AADCC

(c) T-AAD

Fig. 6. Total energy consumption for X-125, AADCC and T-AAD.

[6] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level Sensor Network Simulation with COOJA," in *Proc. of 31st IEEE Conf. on Local Computer Networks (LCN),* , 2006 pp. 641–648.

[7] G. Z. Papadopoulos, J. Beaudaux, A. Gallais, T. Noel, and G. Schreiner, "Adding value to WSN simulation using the IoT-LAB experimental platform" in *Proc. of the 9th IEEE International Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2013.

[8] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks," in *Proc. of the 4th ACM Conf. on Embedded Networked Sensor Systems (SenSys)*, 2006, pp. 307–320.

[9] C. Merlin and W. Heinzelman, "Duty Cycle Control for Low-Power-Listening MAC Protocols," *IEEE Transac. on Mobile Computing*, vol. 9, no. 11, pp. 1508 –1521, nov. 2010.

[10] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *Proc. of the 2nd ACM Conf. on Embedded Networked Sensor Systems (SenSys)*, 2004, pp. 95–107.

[11] C. J. Merlin and W. B. Heinzelman, "Schedule Adaptation of Low-Power-Listening Protocols for Wireless Sensor Networks," *IEEE Transac. on Mobile Computing*, vol. 9, no. 5, pp. 672–685, May 2010.

[12] K.-J. Wong and D. K. Arvind, "SpeckMAC: Low-power Decentralized MAC Protocols for Low Data Rate Transmissions in Specknets," in *Proc. of the 2nd ACM International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality (REALMAN)*, 2006, pp. 71–78.

[13] D. Moss and P. Levis, "BoX-MACs: Exploiting Physical and Link Layer Boundaries in Low-power Networking," *Technical Report SING-08-00, Stanford University*, 2008.

[14] M. Zimmerling, F. Ferrari, L. Mottola, T. Voigt, and L. Thiele, "pTunes: Runtime Parameter Adaptation for Low-power MAC Protocols," in *Proc. of the 11th ACM/IEEE International Conf. on Information Processing in Sensor Networks (IPSN)*, 2012, pp. 173–184.

[15] R. Jurdak, P. Baldi, and C. Videira Lopes, "Adaptive Low Power Listening for Wireless Sensor Networks," *IEEE Transac. on Mobile Computing*, vol. 6, no. 8, pp. 988–1004, Aug. 2007.

[16] A. Meier, M. Woehrle, M. Zimmerling, and L. Thiele, "Zerocal: Automatic MAC Protocol Calibration," in *Proc. of the 6th IEEE Int'l. Conf. on Distributed Computing in Sensor Systems (DCOSS)*, 2010, pp. 31–44.

[17] O. Yang and W. Heinzelman,"Modeling and Performance Analysis for Duty-cycled MAC Protocols with Applications to S-MAC and X-MAC" *IEEE Transac. on Mobile Computing*, vol. 11, no. 6, pp. 905–921, 2012.

[18] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors," in *Proc. of the 29th IEEE International Conf. on Local Computer Networks (LCN)*, 2004, pp. 455–462.

[19] T. Watteyne, D. B. Kris Pister, M. Dohler, and I.A. Blum, "Implementation of Gradient Routing in Wireless Sensor Networks," in *Proc. of the IEEE Global Telecommunications Conf. (Globecom)*, 2009, pp. 1–6.

[20] A. Dunkels, F. Osterlind, N. Tsiftes and Z. He, "Software-based On-line Energy Estimation for Sensor Nodes," in *Proc. of the 4th ACN Workshop on Embedded Networked Sensors (EmNets)*, 2007, pp. 28–32.