Fighting Uncertainty in Highly Dynamic Wireless Sensor Networks with Probabilistic Models

Luís Marques
lmarques@lasige.di.fc.ul.pt
FC/UL*

António Casimiro
casim@di.fc.ul.pt
FC/UL

***Abstract*—Real-time operation in Wireless Sensor Networks (WSNs) is conditioned not only by the current technological level (e.g., limited computing power) but also inherently by the target problem itself: WSNs are required to operate in very open and uncertain environments, subject to external radio interferences, highly dynamic network load, etc.**

Current WSN solutions either provide only best-effort real-time guarantees or make (generally implicit) assumptions on the dynamics of the open environment. These assumptions, in turn, are either very relaxed (i.e., compatible only with undemanding real-time requirements) or very hard to justify.

When dealing with WSNs supporting highly dynamic applications and operating environments (e.g., media streaming, robot control, vehicle coordination, etc.) this problem cannot be ignored. Accordingly, we argue for, and show the efficacy of, using probabilistic models to characterize dynamic WSN QoS, which is the first step to tackle the problem head on. Using our network monitoring technique, we demonstrate that it is possible to meet probabilistic real-time objectives.

***Keywords*-Wireless Sensor Networks; real-time; dependability; adaptation; 802.15.4; non-parametric; lightweight; QoS;**

I. Introduction

Wireless Sensor Networks (WSNs) are used to gather and process information about the state of physical entities in their environment, and may also include control functions. In typical networks the sensor nodes have very limited energy, and therefore the only feasible applications are those where nodes can spend most of their time in low-power sleep modes. These are generally monitoring applications that only infrequently need to send sensor data updates, with transmission periods ranging somewhere between tens of seconds to multiple hours, and correspondingly undemanding deadlines. For these applications it may well be realistically possible to offer even hard real-time guarantees. As long as the WSN has enough redundancy, node faults can be tolerated, while packet transmissions can be retried many times (possibly over redundant network paths) until they succeed, generally well before a timing failure would occur.

We thus understand that, for the currently typical WSN applications, any limitations in real-time guarantees are more a consequence of specific energy budgets than any "intrinsic" inability to offer the desired guarantees. That is, the problem does not derive so much from the open environment the networks operate in, and the uncertainty it implies, but at most from the inability to deal with that uncertainty at a reasonable energy cost. Since we have very loose deadlines and large amounts of redundancy, we should expect that, as long as we were willing to expend energy and we used appropriate techniques, we could recover from faults, be they "internal" (e.g., sensor faults) or "external" (e.g., intermittent wireless interferences), and thus be able to meet even hard real-time objectives.

That is not true for highly dynamic WSNs, such as those employed in teams of cooperating robots sensing their environment. These WSNs or applications thereof involve monitoring and control of very dynamic phenomena in environments where conditions are uncertain and dynamic enough, such that it may not be possible to recover from faults, or from adverse changes in operating conditions, before a timing failure would occur.

For instance, in most WSNs it would likely be unrealistic to assume that a wireless link cannot vary abruptly in quality, or even that it always provides some minimal quality over short timespans. For applications with tight deadlines, such as multimedia streaming, monitoring and control of mobile robots, or coordination of vehicles, this is problematic for a hard real-time design: we cannot dependably design these applications while assuming some static QoS from the network (for some predefined set of resources), and we previously assumed not to have time to recover from faults or QoS changes without incurring in timing failures.

Despite being generally ignored, this problem — that it is not possible to provide hard real-time guarantees in WSNs, in the general case — has been previously recognized in the research community, namely in a recent keynote [1]. But what exactly is the source of the problem? And why can it not be solved by matching the obstacles to real-time operation with the adequate resources and techniques?

The previously referred keynote cited RFC 1925 to wittily motivate the problem: "With sufficient thrust, pigs fly just fine. However, this is not necessarily a good idea." Indeed, there is nothing inherently impossible in meeting hard real-time requirements in WSNs, even demanding ones. Yet, the typically open nature of the environment brings with it the

* This work was partially supported by the EU through the KARYON project (FP7-288195) and the FCT through the Multiannual Funding Program.

possibility of an almost unbounded amount of faults, which in hard real-time designs would have to be dealt with an equally impressive amount of redundancy. For applications with distant deadlines this can probably be achieved with current WSNs, through the exploitation of the large temporal redundancy — at an energy cost, as discussed —, but otherwise this is not practical.

So, our problem is the conjugation of uncertainty from the open environment with real-time requirements that (if strictly interpreted) are incompatible with that uncertainty, under realistic assumptions and acceptable resource usages. Past work has mostly ignored this problem because it is hard to model. Environment uncertainty is (almost by definition) hard to quantify, and its effects are very variable at runtime. Simulations tend to have fairly simplified models, in particular with regards to radio propagation (e.g., circular and symmetric radio coverage models) and external interferences. Therefore, a simulation to show that certain hard real-time requirements can be met under quite specific operating conditions does not address the problem of ascertaining how realistic those conditions are in the first place.

There are two possible solutions. One is to better understand and bound the (currently uncertain) possible operational conditions of these networks. Since the expected trend of WSNs is in the direction of being more general purpose and pervasive, this is probably not a very realistic option. The other solution is to embrace the uncertainty, abandon static environment and fault models, and to use runtime monitoring and adaptation to adapt to actual conditions. This implies that we have to abandon hard real-time models.

In a previous work we explored the possibility of modeling uncertain environments probabilistically, to allow providing probabilistic real-time guarantees, and introduced a technique based on non-parametric statistics [2]. This technique was devised to be computationally lightweight, as would be required for *real-time* monitoring and adaptation in WSNs, but we did not demonstrate its adequacy to WSNs. We only provided a preliminary evaluation of its effectiveness using traces of wired and wireless IP-based networks.

For the adaptation process to be dependable it must be built upon a dependable monitoring technique, which provides an accurate characterization of the current state of the network. Indeed, evaluating an adaptation technique directly can hide problems, as incorrect estimations of the network QoS at time t can be masked by (costlier) adaptations at time $t + 1$. Therefore, in this paper our objective is not detailing how to build an adaptation mechanism but to demonstrate that we can accurately monitor the network QoS.

We show, using a combination of a real network and simulations, that dynamic WSN QoS can indeed be modeled probabilistically. Our monitoring technique, although simple and lightweight, provides a useful assessment of current network latencies, in the form "a latency L can be expected to be met with some probability P", thus providing a dependable foundation upon which to build adaptation mechanisms.

More precisely, the contributions of this paper are:

- Showing that, even in simple and well controlled scenarios, WSNs are significantly affected by interferences, whose bounds are uncertain;
- Demonstrating that our monitoring technique can provide probabilistic estimates of the current network latencies, that are accurate under typical network and environment dynamics (always within ± 6 percentage points of the target probabilities, and generally much closer).

To the best of our knowledge, no previous works have demonstrated a monitoring technique that is sufficiently lightweight to be employed at runtime by the WSN itself and yet dependable enough to support real-time adaptation, so our contribution is of practical interest in addition to theoretical.

The paper is organized as follows. In the following section we better define the problem of achieving (probabilistic) real-time operation in WSNs, and give a quick overview of the applied statistical technique. Then, Section III presents the evaluation scenarios. In Section IV we unveil and discuss the results. In Section V we present related work and Section VI concludes the paper.

II. Probabilistic Real-Time Model and Evaluation Method

WSNs are used to sense and collect the state of physical entities. Applications rely on the WSN to create an internal representation of the state of such entities, to be used in monitoring and control functions. Different applications have different requirements regarding how faithful such representation must be, compared with the true state of the sensed entity.

One of the fundamental factors that affect a reliable perception of the physical environment is the delay between sensing a physical entity and the use of that information by the application. This delay is particularly dependent on the one-way network latency, measured between the sensing node and the sink node, to which the state is propagated. Unfortunately, due to the open nature of the environment over which these networks operate, it is not possible to fully characterize (a priori) the bounds of this latency. As such, we rely instead on a runtime characterization of the network latencies.

We consider that the WSN sends update/event messages at defined intervals, pertaining to the state of a monitored physical entity, which the application uses to create or freshen a corresponding internal representation. According to the dynamics of the monitored physical entity and the acceptable margin of error, the application assumes a deadline until which it is supposed to receive the message. If no message

is received until the deadline a timing failure (at the network level) or fault (at the application level) is considered to have occurred.

The aim of the adaptation process is to limit the occurrence of timing failures. While, due to the open nature of the operating environment, it is not possible to guarantee that timing failures will never occur, their probability of occurrence can be managed. For instance, if the application can change its mode of operation (application-level adaptation) so that it will tolerate a higher margin of error, then the deadline may be extended, increasing the time during which updates may be received and, therefore, tending to lower the probability of timing failures. Another possibility is to change the mode of operation of the WSN itself (network-level adaptation). For instance, by sending duplicate updates through redundant network routes the probability of timely delivery of an update is likely to increase, since radio interferences and other perturbations are often geographically circumscribed. These adaptations generally imply trade-offs; in these examples, lower Qualities of Service provided by the application (due to the increased margin of error) or increased energy and bandwidth consumption (due to the duplicate traffic). Therefore, it is up to the application to define the best trade-off, by indicating the target probability of timing failure.

We achieve probabilistic real-time operation if we dependably maintain an acceptable probability of timing failure. We refer to the expected probability (or observed frequency) of meeting deadlines as the *coverage*, since it denotes how well the application's assumption of timely updates is being covered, and to our objective of achieving probabilistic real-time operation as one of maintaining *coverage stability*.

In this paper we analyze how well such coverage stability can be achieved in 802.15.4-based WSNs without using network-level adaptation. We do this by continually monitoring network conditions and testing how well the estimated conditions hold (which is equivalent to application-level adaptation) despite the environment dynamics. This tells us how accurate our monitoring is, and thus allows us to evaluate if we have a dependable monitoring solution upon which to build complex adaptation mechanisms.

Network Monitoring

We consider as the monitoring target the network's one-way latency that occurs between the beginning of sending the update/event message and its reception at the sink. One such interval is exemplified in Figure 1, identified as "message latency". It differs from the "packet latency" in the case when packets are lost. Newer updates are considered to supersede old ones, which are discarded if received out of order.

The message latency can be measured if a notion of global time is assumed (there are several clock synchronization algorithms optimized for WSNs [3][4]) and the messages

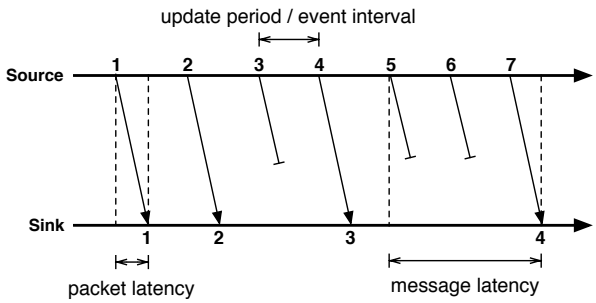

Figure 1: Timing variables (example)

are timestamped before being sent. Otherwise it may be estimated by other methods; for instance by periodically measuring round-trip times.

At the sink the latencies are computed (against the local synchronized clock) and collected, up to a given sample size n, with old sample values being discarded.

Adaptation

Our monitoring evaluation is equivalent to an application adapting to the perceived network conditions (but not the network). At each instant we have some assumed latency, which we expect to be met with some chosen probability (see the evaluation subsection). When we receive a new message we test if the assumed latency was met or not. We also make a new estimate of the latency for the chosen probability, and update our assumption (i.e., the application would adapt to the new latency). At the end of the trial we compute the average coverage, by averaging the number of met deadlines over the total number of messages.

Probabilistic analysis

The non-parametric analysis upon which the adaptation is based is simple and efficient. A copy of the sample is ordered. The first value (the 1^{st} order statistic) is the lowest latency observed, and if it were used as a deadline we would expect a high number of timing failures, since most subsequent latencies would likely be higher. Conversely, the last value of the ordered sample (the n^{th} order statistic) is the highest observed latency, and therefore most subsequent latencies would likely be lower. The previously proposed non-parametric adaptation algorithm [2] very efficiently chooses the latency value given by the order statistic k which best matches the target coverage C, based on the following equation:

$$k = C * (n + 1) \tag{1}$$

This method was chosen because it is very efficient, as required to be executed continually at runtime by the WSN nodes themselves, yet provides an accurate probabilistic view of the latencies that can be used as the basis for complex adaptation mechanisms.

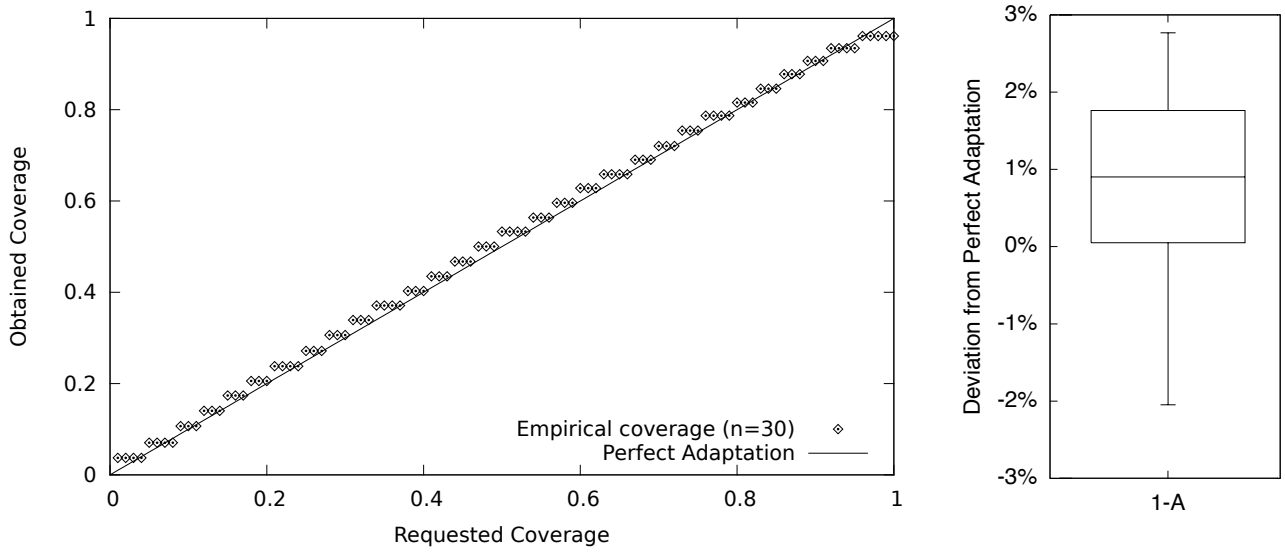

Figure 2: Example of summarizing coverage deviations

Evaluation

According to the relied upon statistical properties, if the WSN never changed between the instant at which the latencies were characterized and the deadline instant, then the observed frequency of timing failures would tend toward the expected value (the mean), with negligible deviations for long-running applications. In this paper we measure the observed deviations from the target coverages to evaluate how robust our monitoring technique is under typical WSN dynamics (i.e., how well our stability assumption holds [2]).

Because the impact of latency variations can depend on the probability with which we want to meet deadlines, we test a wide range of coverages. For instance, if the observed latencies are bimodal and they change in such a way that only one of the two modes varies, then that change will have no impact on adaptations based only on the other mode. As such, for a full characterization of the impact of environment dynamics, we test all coverages between 1% and 99%, in percentage point increments.

To illustrate the deviations between the requested and the achieved coverage for a single scenario we can create a graphic plotting all tested target coverages and the associated empirically obtained coverages. As an alternative, we can summarize the deviations in the form of a box plot graphic, which condenses the information and facilitates comparison of alternative scenarios. Figure 2 exemplifies this process. In this paper we used such box plots[1].

In our example, we observe that the obtained coverages are generally slightly higher than the requested coverages (i.e., they are above the perfect adaptation line). Accordingly, the box plot summarizes this by showing that most (about 75%) of the population (i.e., most of the tested coverages) is above the 0% deviation from perfect adaptation. No outlier (represented as a dot) is present in the box plot because no observed coverage deviated enough (about 3 percentage points from the lower or upper quartile, in this example).

[1]The whiskers are defined to represent, as is common, the most extreme values which still lie within 1.5 times the interquartile range from lower and upper quartile (or the minimum and maximum values, if no outliers are present)

III. Evaluation Scenarios

In this paper we present results for a small real network, as well as various scenarios which were simulated with the omnet++ simulator using the inetmanet package. All networks are based on a 802.15.4 PHY/MAC stack (which is a good trade-off between supporting highly dynamic applications and still conserving some energy). These simulation scenarios are structured as three base scenarios (1, 2, 3), with variants (e.g., 1–A, 1–B, etc). In both the real network and in the simulations we used only a non-beacon mode, with CSMA/CA, to make the network dynamics more easily visible and because we believe that not relying on a centralized beacon would more appropriately reflect the kind of large-scale networks that are envisioned in the literature for the future [5]. We chose the end-to-end latency as the variable used for monitoring/adaptation. Nodes send an acknowledgement (ACK) upon successful frame reception, as specified in the 802.15.4 standard. Periods of 600 seconds of activity were analyzed.

The use of a real network allows us both to (1) show the impact of the uncertain interferences, and (2) to evaluate the monitoring accuracy under real (if simple) conditions, while the simulations allow the evaluation of more complex scenarios. The real network and the simulation base scenarios 1 and 2 were designed to be (as much as possible) compa-

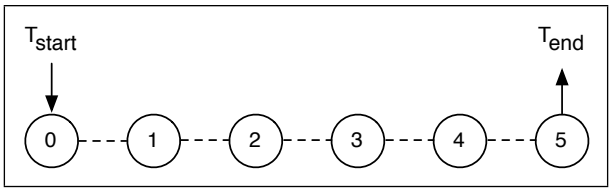

Figure 3: Simulation base Scenario 1 (1–A & 1–B)

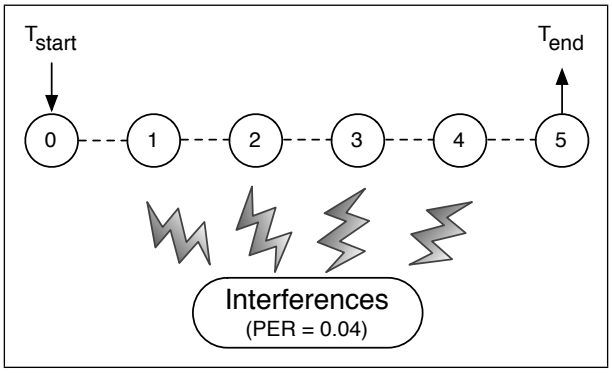

Figure 4: Simulation base scenario 2

rable. This is helpful, since by comparing real results with simulation results we can better understand how realistic and meaningful the simulation results really are.

The simulation base scenarios 1 and 2 and the real network scenario follow a similar structure: all nodes are arranged in a logical line, with nodes only communicating with their immediate neighbors. The first node produces the updates, the last node is the sink to which the updates must arrive, and the intermediate nodes forward these updates.

In the simulation base scenario 1 (see Figure 3), neither movement nor any kind of interferences are introduced, so the variations in end-to-end latency are expected to reflect factors which are intrinsic to 802.15.4, such as the CSMA/CA exponential back-off. The variants 1–A and 1–B have 6 nodes, to allow comparisons with the real network, while the variant 1–C tests the impact of varying the number of nodes. We also test different traffic patterns and different sample sizes. The traffic patterns chosen were a Constant Bit Rate (CBR), to model updates at periodic intervals (e.g., media streaming), and a Poisson / Exponential pattern, to model the detection of natural events, which do not occur at fixed periods (e.g., moving obstacles). Short update intervals were used, reflecting our focus on highly dynamic WSNs and applications.

The simulation base scenario 2 (see Figure 4) introduces a source of interferences, producing a packet error rate of 4%, which is intended to mimic the existence of inferences in real networks operating in open environments.

The real network (see Figure 5) was built using six RCB128RFA1 V6.3.1 evaluation modules from dresden elektronik. These boards contain an AVR ATmega128RFA1

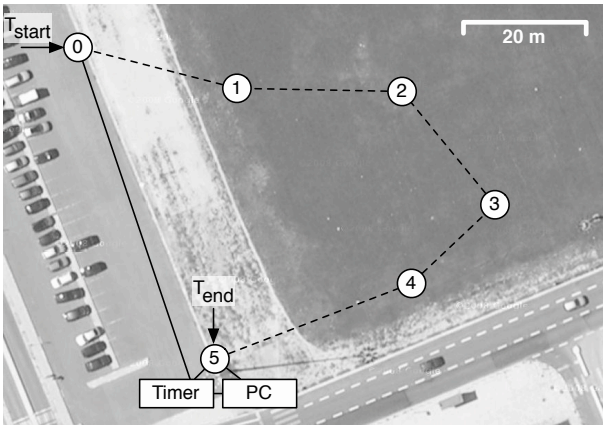

Figure 5: Real network scenario

chip, which integrates an ATmega1281 microcontroller with an AT86RF231 transceiver, compatible with 802.15.4 at 2.4 GHz. The nodes were configured to use the minimum transmission power, by setting the PX_PWR3:0 bits of the PHY_TX_PWR register to one, resulting in a -16.5 dBm power output (0.0224 mW).

As much as possible, we tried the real network to be conceptually equivalent to a (logical) linear network. Two major factors decided the actual geographical distribution of the nodes, which is illustrated in Figure 5.

One factor was that, instead of using a clock synchronization algorithm, we measured the end-to-end latency through wired electronic signals, so we could not have the source and the destination nodes too far apart. Such setup allowed us to better isolate any variation in the latencies, and thus be sure our results reflected actual network and environment dynamics. This was performed by having the source node (node 0) and the sink node (node 5) signal in microcontroller pins when the transmission started and when the update message was received. These signals were timed with real-time equipment, with 16 μs of accuracy, and forwarded to a PC for storage and analysis. The sink node also connected to the PC to share the sequence number of the messages received, so that in cases of multiple consecutive lost messages no ambiguity would arise.

Another factor was the land relief. Although the terrain was fairly flat, we observed that nodes had to be placed at different distances among each other to achieve equivalent signal quality. We tried to obtain a node disposition which best increased signal quality among neighbor nodes and decreased interference among unconnected nodes, which differed significantly from the planned arrangement. We observed that nodes which should ideally not be connected nor interfere among each other had to be within distances where sometimes (but rarely) was still possible to communicate, in order to achieve good link quality between the connected

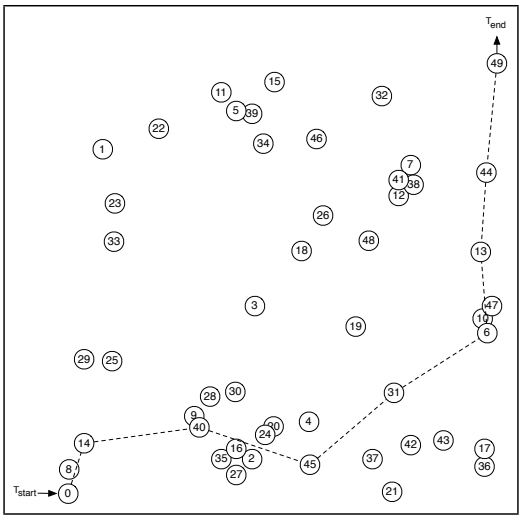

Figure 6: Simulation base scenario 3 (Example)

nodes. This also exemplifies how uncertain the properties of real WSN deployment environments really are.

The real network scenario has similarities with both the simulation base scenarios 1 and 2. While we performed our evaluation in a geographical area and during a time of the day where low 2.4 GHz WLAN activity was present, there is always some amount of interferences. These include thermal noise, microwave ovens, 802.11 WLANs and the non-neighbor network nodes. Therefore, we would expect results from the real network to be somewhere between those of simulation scenarios 1 and 2.

The simulation base scenario 3 departs from the other scenarios to implement a more complex network, as one would expect to find in a real WSN (see Figure 6). It is less artificial, and does not isolate as well the dynamics of the 802.15.4 physical and MAC layers, but allows studying the impact of internal network dynamics, such as node mobility.

In the simulation scenario 3–A the nodes are randomly dispersed in a 50×50 meter area, except for the source and sink nodes which always start at opposing edges. The traffic is routed through the intermediate nodes using the Ad hoc On-Demand Distance Vector Routing (AODV) protocol. The variant 3–B changes the traffic pattern from constant to random with an exponential probability distribution, while the variant 3–C introduces movement of the nodes and 3–D further adds four competing traffic flows. Scenario 3–D is therefore the most complex, and is a good test to evaluate whether the monitoring is accurate, even in the presence of highly challenging dynamics.

These scenarios and the analyzed variants are summarized in Table I. Other variants (e.g., different update intervals or sample sizes) were explored, but only those which provided important insights on the limitations of monitoring effectiveness are here presented and discussed.

Scenario	# Nodes	Update Interval	Sample Size (n)
Real Netw.	6	Constant (0.05 s)	20, 30, 60, 100
1–A	6	Constant (0.05 s)	20, 30, 60, 100
1–B	6	Exp. ($\lambda^{-1} = 0.05$)	30
1–C	5–30	Constant (0.05 s)	30
2	6	Constant (0.05 s)	30, 100
3–A	50	Constant (0.05 s)	30
3–B	50	Exp. ($\lambda^{-1} = 0.05$)	30, 100
3–C (M)	50	Exp. ($\lambda^{-1} = 0.05$)	30
3–D (M+C)	50	Exp. ($\lambda^{-1} = 0.05$)	30, 100

Key
(M): movement of nodes
(C): competing message flows

Table I: Overview of evaluation scenarios

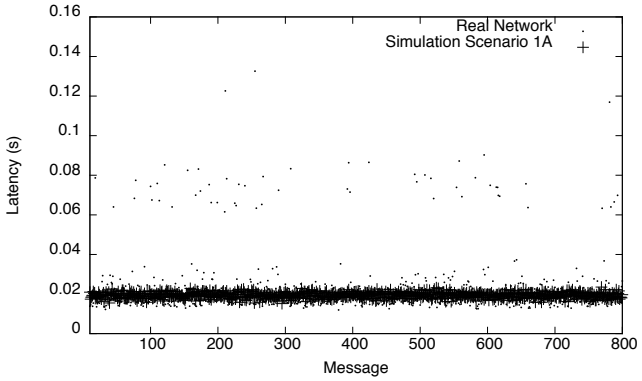

Figure 7: Latencies: real network vs 1–A

IV. Results

Figures 7 and 8 compare the end-to-end message latency of the simulations versus the empirical results of the real network. The latencies of the first 800 messages of each are plotted.

As we can see in Figure 7, in both the simulation and the real network most latencies are distributed around 20 ms, with some random dispersion. The dispersion is due to mechanisms such as the 802.15.4 CSMA/CA during the contention access period (CAP) and the MAC-level transmission retries — up to 7 retries are allowed by the 802.15.4 specification, the default of 3 maximum retries was used in both the real network and the simulations. The simulation has a slightly smaller dispersion (as might be expected, since the simulation does not reflect all realistic sources of interference), but is otherwise very similar (the same scenario simulated using ns-2 showed greater divergence). Because no channel interferences were introduced, almost no messages were lost during the whole simulation period, despite the high update data rate. As such, the simulation does not exhibit latencies around 70 ms (1 lost update) and 120 ms (2 lost updates) like the real network does.

The simulation scenario 2 more closely matches the empirical results of the real network, as is visible in Figure 8. Due to the introduction of interferences, packets were

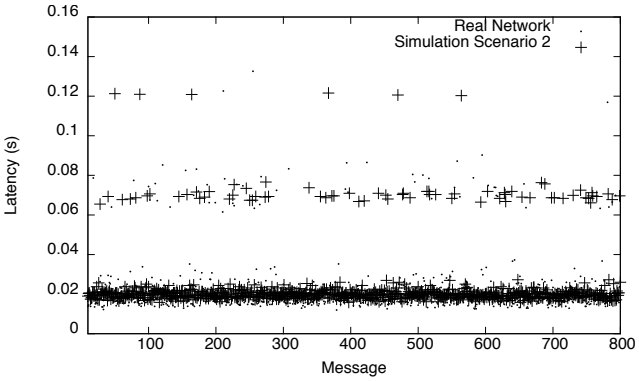

Figure 8: Latencies: real network vs 2

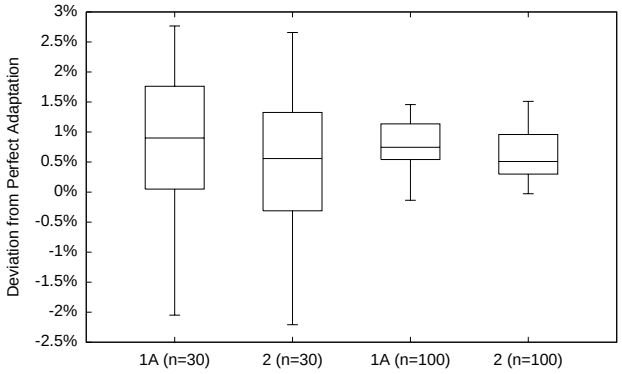

Figure 9: Scenario 1–A vs scenario 2

dropped, creating a probabilistic pattern of three different latency bands, caused by application-level "retries" (superseding updates), with a progressively lower occurrence of latencies in each band as the number of retries increases.

There are three important conclusions regarding these results. The first is that, despite the real network being very simple and small, and despite our efforts to reduce all sources of interference as much as possible, we needed to include an explicit source of interferences in the simulation to accurately replicate the empirical results. This stresses the real impact of the open environment and of complex radio interference patterns, which are very hard to characterize and bound for all realistic scenarios that a real network might face. The second is that our dispersion of latencies is indeed quite random, and thus compatible with the assumptions of our probabilistic approach. The third is that the simulation results agree well with those of the real network, which establishes a baseline of trust in the results from the other simulations we present.

We can make a first assessment regarding the monitoring effectiveness in small and linear simulated networks by inspecting Figure 9. We can note that the coverage (i.e., the inverse of the observed frequency of timing failures) never exceeded the target by more than 3 percentage points, nor ever underachieved it by more than 2.5. Thus, as preliminary finding we could expect that: (1) even without using network-level adaptation to meet QoS targets, our monitoring solution would be accurate enough to support application-level adaptations that should come within about 3 percentage points of the best performance/timeliness trade-off for even the most adverse coverage; (2) that when also using network-level adaptation we would only have to expend a small amount of resources to compensate for the slightly inaccurate QoS estimates.

We also notice that despite scenario 2 introducing interferences which are not present in scenario 1–A, the monitoring effectiveness is not worse in scenario 2, and even improved slightly (the median is closer to the target).

Another preliminary conclusion that can be taken from Figure 9 is that larger sample sizes (increased n) tend to lead to more accurate latency estimates, since the box plots for the n = 100 have a smaller amplitude than those for n = 30, reflecting more accurate monitoring.

From these results we can tentatively conclude that increasing the complexity of scenarios (i.e., introducing factors of temporal uncertainty) does not in itself create an impediment to obtaining accurate monitoring. On the contrary, we observe that the results may even improve. The explanation for this outcome may be understood by analogy to the central limit theorem (CLT) from probability theory. In the CLT, as we add an increasing number of independent random variables (with finite mean and variance) the average starts to approximate a normal distribution, even if the variables were not normally distributed. Likewise, as we add various factors which affect the latency in unpredictable ways the result may start to be, overall, more predictable.

A similar effect can be seen in Figure 10, which compares the scenario 1–A (a constant rate update stream) with scenario 1–B (updates at exponentially distributed intervals). While the amplitude of the box plots for these scenarios is similar, the median deviation from the target coverage for the scenario 1–B is much closer to ideal of 0 percentage points.

Figure 11 more comprehensively explores the effect of increasing the sample size, showing the summarized results of scenario 1–A. We see that by gradually increasing the sample size we gradually improve the monitoring accuracy, although with progressively smaller gains. This diminishing of returns is not unexpected, since the dynamics of the WSN can endanger the freshness of the older sample values, counterbalancing their benefit for assessing the current state of the network's QoS.

Figure 12 presents a similar analysis, but performed using the latencies of the real network. The most important conclusion that should be taken from Figure 12 is that the monitoring is highly effective under a real scenario, with

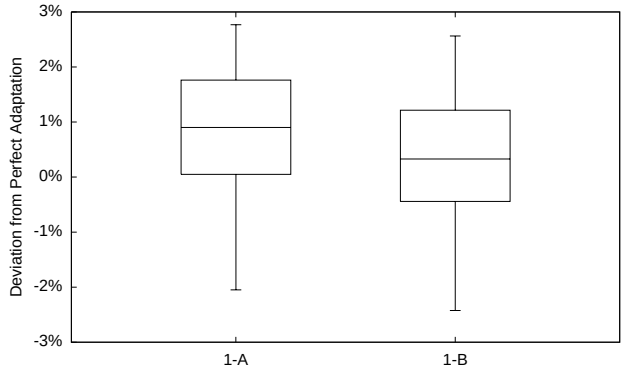

Figure 10: Scenario 1–A vs 1–B

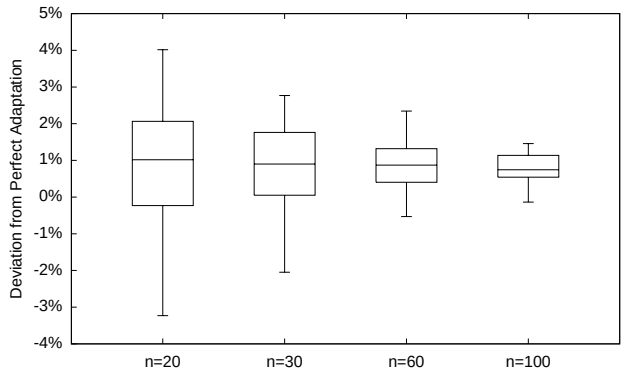

Figure 11: Scenario 1–A — Adaptation effectiveness at different sample sizes (simulated)

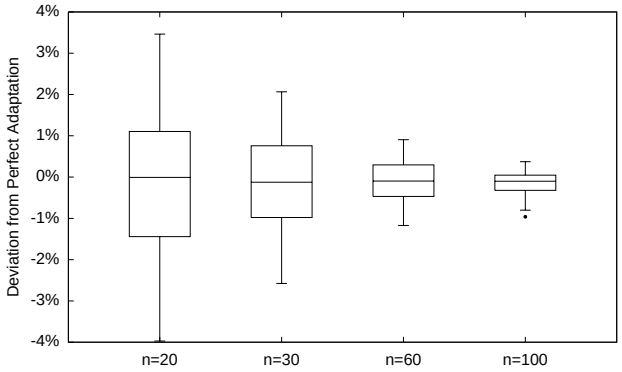

Figure 12: Real Network — Adaptation effectiveness at different sample sizes

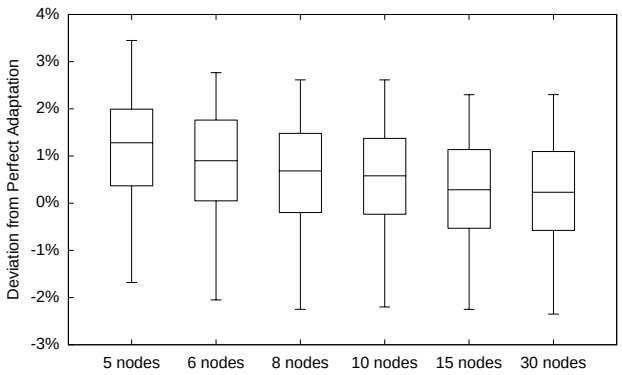

Figure 13: Scenario 1–C — Effect of number of hops in adaptation effectiveness (simulated)

small measured deviations from the target coverage. First, even for the small sample sizes of $n = 20$ and $n = 30$ we obtain good results, even better than those obtained under simulation with scenario 1–A. Second, by increasing the sample size we are able to come very close to the target coverages; for $n = 100$ we always come within less than 1 percentage point, which implies that our monitoring strategy estimated very accurate latencies for all target coverages. Third, there is slightly less of an effect of diminishing returns, compared with simulation scenario 1–A. This is most likely due to the increased randomness of the real network, compared with the simulation.

Another important consideration when exploring the dependability of probabilistic QoS models is the existence of scale effects. To address this question, Figure 13 scrutinizes the impact to the adaptation of the number of hops that must be traversed between the update/event producing node and the sink node. The number of nodes of a linear network (scenario 1–C) is increased between 5 and 30, corresponding to 4 and 29 hops.

We see in Figure 13 that not only does the adaptation not degrade with the increase in the number of hops that need to be transversed but that the median deviation actually improves. That is, while the average latency will increase as the number of hops rises we can still effectively estimate probabilistic bounds for dynamic latencies. This result strongly supports our hypothesis that the proposed monitoring strategy can dependably support adaptation mechanisms for probabilistic real-time operation in real networks, under a variety of different circumstances; from small networks to large networks.

While the simplicity of the simulated scenarios 1 and 2, as well of the real network, provide good testing grounds for a variety of different experiments (which highlight particular properties and important corner cases), they do not reflect the complexity of realistic larger-scale networks. The base simulation scenario 3 addresses the effectiveness of monitoring in more complex and realistic networks.

The four concrete scenarios derived from the base scenario 3 address increasingly complex situations. First we start with static nodes which communicate at a constant rate (scenario 3–A). Then we change to a probabilistic interval model (scenario 3–B). To this we add the movement of network nodes (including the source and sink nodes — scenario 3–C) and, finally, we incorporate competing message

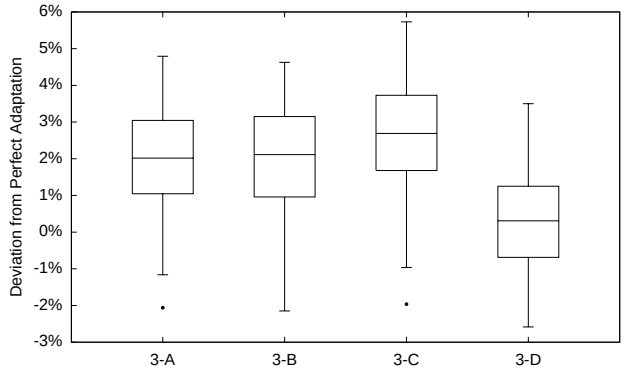

Figure 14: Scenarios 3–A, 3–B, 3–C and 3–D

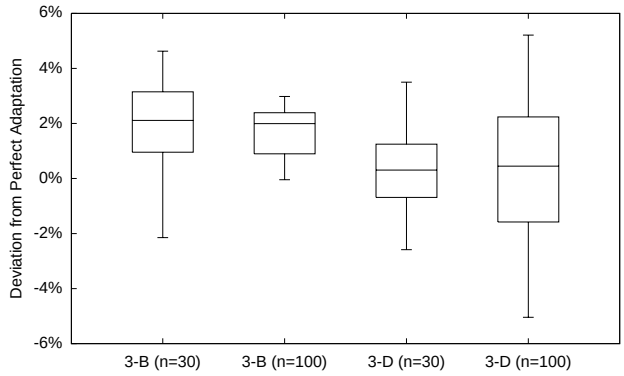

Figure 15: Scenario 3 — Adaptation effectiveness at different sample sizes

flows, which will further disrupt the latencies pertaining to the measured flow of event update messages (scenario 3–D). Figure 14 presents the adaptation results for all of these scenarios.

In general, we observe that even under complex scenarios, and with a small sample size (n = 30), our monitoring strategy is still fairly accurate, as all of the simulated scenarios never deviate more than +6 or -3 percentage points from the target coverage, and in most cases quite less.

More particularly, we notice that, as happened in the real network and in the simulated base scenario 1, the transition from a constant (scenario 3–A) to a random message interval (scenario 3–B) did not disrupt the monitoring. Comparing the box plots for scenarios 3–B and 3–C we witness a very slight decline in monitoring accuracy, as movement is introduced in the network. Contrarily, introducing competing messages flows on top of node movement, as was done in scenario 3–C, again improved the monitoring accuracy, with lower deviations from the target coverage.

Finally, we present and discuss the effect of increasing the sample size in the more realistic base scenario 3. Figure 15 shows the results.

We see that when no movement was present (scenario 3–B) the increase in sample size produced more consistent monitoring results, with less variation in the deviations. On the other hand, when in the presence of movement (scenario 3–C) the increased sample size worsened the monitoring effectiveness. This makes sense, and also helps explain the behaviors exhibited in Figure 15. Due to the dynamics created by the node movement, older sample values no longer accurately reflect the current state of the network, making for less accurate estimates of the latency that matches the chosen coverage.

V. Related Work

One of our long-term research objectives is to improve the dependability of real-time WSNs. Therefore, our work is in general related to real-time and QoS provision in the context of WSNs, but more specifically with mechanisms and approaches to deal with the uncertainties affecting the timeliness of communication.

Several works address the problem of providing QoS and real-time guarantees in WSNs, which can be tackled at different levels, including the network architecture, sensor node hardware, and protocols at the different layers of the communication stack (MAC, network, transport, application) [6]. One particularly relevant and important approach for dependable and real-time operation is to exploit the intrinsic redundancy existing in large-scale WSNs. While the deployment of many nodes can be a cause of interferences and increased uncertainty, it also allows for multi-path protocols to be considered [7]. A different approach is to differentiate classes of traffic, assigning higher priorities to urgent or real-time data. For instance, the RAP architecture [8] takes into account routing distances when assigning priorities and routing packets, to reduce deadline misses for packets far-away from their destination.

One of the fundamental problems for predictability and dependability is the interference that derives from the crowded electromagnetic spectrum [9]. In this paper we considered several scenarios involving interference, to observe the behavior of communication from a temporal perspective, and to investigate the possibility of correctly characterizing this behavior with our probabilistic analysis approach. The work presented in [10] also considers the impact of interference processes in WSNs. However, their goal is to characterize the interference itself rather then its effects on timeliness, and to propose solutions to reduce or avoid this interference, and thus achieve a better overall behavior. The detection of radio interferences, as done in [11], can be relevant from a communication reliability perspective, while the possibility of assessing the wireless link quality [12] can, in general, be useful for achieving dependable adaptation.

There is competing work in probabilistic models of network QoS [13] but not with a focus on being lightweight (and thus allowing runtime monitoring from the network

nodes themselves), with estimations taking several seconds on a modern computer.

Finally, experimentally observing, measuring and characterizing the behavior of WSNs is not easy, as it usually involves the need of special setups and measurement equipment [14]. Our approach was to use additional measurement equipment (a dedicated network node, plus a PC) to perform these measurements. On the other hand, experiments are usually done for specific application environments (e.g., industrial ones [15]) or using existing testbeds (e.g., one that was previously used to validate a routing protocol [16]), whereas our objective in this case was to create scenarios that would allow us to compare the results of a real setting with those of simulated systems.

VI. Concluding Remarks

Dependably achieving real-time operation in wireless and open environments is a hard problem. The nature of open environments makes hard real-time guarantees either extremely pessimistic or unreasonably expensive, and further makes this a hard issue to research, since it is not clear what typical bounds can be safely assumed. Indeed, our results showed that in real networks, even in simple and well controlled scenarios, interferences have a significant impact on the results.

We argued that an approach based on runtime monitoring and adaptation can overcome this limitation, and that establishing the dependability of a monitoring technique such as ours is an essential step to take before building complex adaptation mechanisms, which might hide inaccurate monitoring estimates, at a steep cost.

While in an open environment we cannot assume any strict bounds of the network and environment dynamics, we saw that under typical conditions our monitoring approach accurately estimates the current network conditions, providing the latencies that should be assumed to meet deadlines with the desired probability. In particular, we saw that the estimates allowed us to always come within ± 6 percentage points of the desired timeliness probabilities, and generally much closer. This monitoring approach thus provides a solid base upon which to build dependable adaptation techniques.

References

[1] Decotignie, J.D.: The real-time and wireless sensor networks: are they compatible? In: 24th Euromicro Conference on Real-Time Systems, ECRTS 2012. (2012) Keynote address.

[2] Marques, L., Casimiro, A.: Lightweight dependable adaptation for wireless sensor networks. In: Proceedings of the 30th IEEE International Symposium on Reliable Distributed Systems Workshops, 4th International Workshop on Dependable Network Computing and Mobile Systems (DNCMS 2011), Madrid, Spain (oct 2011) 26–35

[3] Li, Q., Rus, D.: Global clock synchronization in sensor networks. IEEE Transactions on Computers **55**(2) (2006)

[4] Yoon, S., Veerarittiphan, C., Sichitiu, M.L.: Tiny-sync: Tight time synchronization for wireless sensor networks. ACM Trans. Sen. Netw. **3**(2) (June 2007)

[5] Corke, P., Wark, T., Jurdak, R., Hu, W., Valencia, P., Moore, D.: Environmental wireless sensor networks. Proceedings of the IEEE **98**(11) (nov. 2010) 1903 –1917

[6] Akyildiz, I.F., Melodia, T., Chowdhury, K.R.: A survey on wireless multimedia sensor networks. Computer Networks **51** (2007) 921–960

[7] Li, S., Neelisetti, R., Liu, C.: Efficient multi-path protocol for wireless sensor networks. International Journal of Wireless and Mobile Networks (IJWMN) (Jan 2010)

[8] Lu, C., Blum, B., Abdelzaher, T., Stankovic, J., He, T.: Rap: A real-time communication architecture for large-scale wireless sensor networks. In Real-Time Technology and Applications Symposium (2002)

[9] Zhou, G., Stankovic, J.A., Son, S.H.: Crowded spectrum in wireless sensor networks. In: Proceedings of Third Workshop on Embedded Networked Sensors (EmNets. (Jan 2006)

[10] Liang, C.J.M.: Interference characterization and mitigation in large-scale wireless sensor networks (2011)

[11] Zhou, G., He, T., Stankovic, J.A., Abdelzaher, T.: Rid: Radio interference detection in wireless sensor networks. In: in INFOCOM. (2005)

[12] Baccour, N., Koubía, A., Ben Jamía, M., do Rosário, D., Youssef, H., Alves, M., Becker, L.B.: Radiale: A framework for designing and assessing link quality estimators in wireless sensor networks. Ad Hoc Netw. **9** (September 2011) 1165–1185

[13] Wang, Y., Vuran, M.C., Goddard, S.: Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. IEEE/ACM Trans. Netw. **20**(1) (February 2012) 305–318

[14] Ageev, A., Macii, D., Petri, D.: Experimental characterization of communication latencies in wireless sensor networks. In: Proceedings of the 16th IMEKO TC-4 International Symposiu "Exploring New Frontiers of Instrumentation and Methods for Electrical and Electronic Measurements" and 13th Workshop on ADC Modelling and Testing, IMEKO, A&T (2008) 258–263

[15] Bertocco, M., Gamba, G., Sona, A., Vitturi, S.: Experimental characterization of wireless sensor networks for industrial applications. Instrumentation and Measurement, IEEE Transactions on **57**(8) (aug. 2008) 1537 –1546

[16] Pavkovic, B., Theoleyre, F., Barthel, D., Duda, A.: Experimental analysis and characterization of a wireless sensor network environment. In: Proceedings of the 7th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks. PE-WASUN '10, New York, NY, USA, ACM (2010) 25–32