Extracting Markov Chain Models from Protocol Execution Traces for End to End Delay Evaluation in Wireless Sensor Networks

François Despaux, Ye-Qiong Song, Abdelkader Lahmadi

Extracting Markov Chain Models from Protocol Execution Traces for End to End Delay Evaluation in Wireless Sensor Networks

François Despaux
LORIA, Université de Lorraine
Vandoeuvre-lès-Nancy, 54500, France
Email: francois.despaux@loria.fr

Ye-Qiong Song
LORIA, Université de Lorraine
Vandoeuvre-lès-Nancy, 54500, France
Email: song@loria.fr

Abdelkader Lahmadi
LORIA, Université de Lorraine
Vandoeuvre-lès-Nancy, 54500, France
Email: abdelkader.lahmadi@loria.fr

***Abstract*—Many WSN industrial applications impose requirements in terms of end to end delay. However, the end to end delay estimation in WSNs is not a simple task because of the high dynamic of networks, the use of duty-cycled MAC protocols as well as the impact of the routing protocols. Markov-based modelling is an interesting approach to deal with this problem aiming to provide an analytical model useful for understanding protocol's behavior and to estimate the end to end delay, among other performance parameters. However, existing Markov-based analytic models abstract the reality simplifying the analysis and thus resulting models are not accurate enough for estimating the end to end delay. Furthermore, establishing an accurate Markov model using classic approaches is very difficult considering the highly dynamic behavior of the sensor nodes. In this paper, we propose a novel approach to obtain the Markov chain model of sensor nodes by means of Process Mining techniques through the code execution trace. End to end delay is then computed based on this Markov chain. Experimentations were done using IoT-LAB testbed platform. Comparisons in terms of delay are presented for two different metrics of the RPL protocol (hop count and ETX).**

***Keywords*-MAC Protocols; Markov chain; Process Mining; Wireless Sensor Network performance.**

I. Introduction

Understanding the behavior and limitation of wireless sensor networks (WSNs) is important for estimating performance metrics such as end to end (e2e) delay, throughput, energy consumption, etc. Consequently, modelling the behavior of the networks becomes essential for estimating these metrics, in particulary the e2e delay. Many WSN industrial applications such as safety system, control systems and monitoring systems impose a requirement on packet delay. In monitory systems for instance, time-critical alarm packets may impose bounds in terms of delay from the moment an event is detected until the information reaches the destination. Different approaches have been proposed in literature for modelling the network behavior, including analytical modelling and simulation based analysis. Due to its high dynamic nature, WSNs present a number of challenges which do not exist, or exist in rather different forms, in traditional wired networks. Therefore, modelling the behavior of such networks is challenging and not a straightforward task. Normally, proposed models abstract the reality simplifying the analysis and therefore, resulting models are not accurate enough. Hence, estimating the e2e delay in a precise manner becomes difficult. As an example of this, the widely spread standard IEEE 802.15.4 MAC protocol has been extensively analyzed in the literature. In [9], authors present a Markovian model from which expression for the one hop delay distribution are found. However, the reality was simplified by assuming a buffer size of one packet on each node, an assumption that do not represent the reality of a sensor node. A more complete model for this protocol was proposed by Misic et al. in [8] where many aspects of the protocol such as duty cycle and finite buffer size were considered and where nodes were modeled as an M/G/1/K queuing system. However, the model lacks of a realistic radio channel model and capture effect model. Besides, neither [9] nor [8] consider multi-hop tranmission scenarios. Two well-known alternatives of analytical models are simulations and testbed or field measurements. There are many tools for simulating WSNs but it is still difficult to take into account some network and implementation details (e.g. capture effect and the impact of OS). Measurements on actual networks can effectively capture all these features. Moreover, by successives measurements, we can obtain the e2e delay distribution curve. However, we cannot obtain a mathematical expression of the e2e delay distribution. Finding this analytical expression will allow us to calcule the probability of deadline miss, for instance, the P[delay $\leq$ deadline]. The analytical e2e delay distribution allows to easily get the probabilistic delay guarantee. In this paper we propose a novel approach combining measurement-based and analytic approaches for finding this mathematical expression. The main difference between our proposed approach and measurements is that, even if our approach is in some way measurement-based, we are able to obtain an analytical expression of the e2e delay distribution. Previously in [3] and [4], we have shown the suitability of our approach for modelling the behavior of both IEEE 802.15.4 and ContikiMAC MAC protocols and we explained how the e2e delay in a WSN can be estimated. However, both [3] and [4] consider static routing, that is to say that no dynamic

routing strategy is considered (packets are sent from source to destination always following a predefined path). In this paper, we go one step beyond by considering a new MAC protocol and also a routing protocol (RPL) running on each node. Our objective then is to extend the analysis done in [3] and [4] considering a dynamic routing protocol to see whether the approach is still suitable for modelling the network behavior in a more realistic scenario. Furthermore, since RPL considers two routing strategies based on two objective functions, we make use of the obtained model to present a performance comparison between these two strategies in terms of the e2e delay distribution and packet reception rate. Finally for computing the e2e delays in a large scale multi-hop transmission scenario, this approach bypasses the difficulty of modelling the input flows of the forwarders (generally not Poisson arrivals), by directly using the Markov chain of those nodes. Our main contribution is a new approach allowing to obtain the e2e delay expression in WSNs running both dynamic MAC and routing protocols. To our best knowledge, there is no previous work that can provide it. The reminder of this paper is organized as follows. Section II presents main related work on analytic modelling of duty-cycled MAC protocols. Section III gives a background of the X-MAC and RPL protocols, as well as the process mining approach. Our combined measurement-analytic methodology is presented in Section IV. Samples of results are presented and discussed in Section V. Finally Section VI gives concluding remarks and outlines future work.

II. Related Work

In this section we only focus on the review of the main analytic models of MAC protocols for WSNs, most of them are based on Markov chain modelling and are developed for the IEEE 802.15.4 MAC protocol. Only a few work can be found for other MAC protocols such as S-MAC and X-MAC. In [11], authors propose a Markov queuing model to analyse the throughput, delay and energy consumption of both synchronized and asynchronous duty-cycled MAC protocols. By means of this model, they analyse the behavior and the performance of both S-MAC and X-MAC. However, the model does not consider retransmissions, the channel is ideal (no fading and no capture effect) and the model works for deterministic channels (constant packet loss probability). Moreover, the performance analysis is focused on one node so the delay analysis only considers one-hop transmission scenario. A survey on latency issues of duty-cycled MAC protocols is provided in [5]. Authors provided expressions for both one-hop and e2e delay. Although they can be used to estimate the protocol efficiency in terms of delay, these expressions for the one-hop delay do not consider the queuing delay, which is an important component that impacts in the whole e2e delay. Besides, expressions for the e2e delay assumes that there is a single traffic in the network, limiting thus its practical use. As far as MAC protocol is concerned, the existing models focus on the single node behavior. Most of the proposed solutions for IEEE 802.15.4 are based on Markov models initially developed for IEEE 802.11 standard. These models have been extended for modelling the IEEE 802.15.4 MAC protocol under different assumptions. Misic et al. [8] proposed a Markov chain model for the standard IEEE 802.15.4 MAC protocol considering a $M/G/1/K$ queue model and superframe with both active/only and active/inactive duty-cycle periods for a star topology (one hop). Expressions for the access delay, probability distribution of the packet service time as well as probability distribution of the queue length are presented. The limitation of this model is that all results were obtained for 1-hop transmission where a device sends a packet to a coordinator and waits for the acknowledgement. Besides, even considering a $M/G/1/K$ queue system for the first node, and since the output distribution of a $M/G/1/K$ is not necessary Markovian, it is not possible to extend the proposed model for multi-hop transmissions by chaining $M/G/1/K$ queue system. Instead, a $M/G/1/K \rightarrow G/G/1/K \rightarrow G/G/1/K \cdots G/G/1/K$ queue system must be considered. However, modelling this kind of queuing systems is not straightforward. RPL protocol has been extensively studied in the literature. In [1], authors analyse RPL together with the underlying X-MAC protocol from the reliability stand point. Finally, authors in [7] developed a theoretical framework to estimate the end-to-end delay in a networked system using frequency-domain modelling and analysis where they have shown that their approach is more scalable and allows analysis of compositional networked systems. In this paper, we apply this methodology to compute the e2e delay.

III. Background

In this section, we give an introduction to both X-MAC and RPL protocols. Details about the process mining algorithm can be found in [3] and [4].

A. X-MAC

Duty-cycled MAC protocols can be roughly categorized into synchronous and asynchronous. The idea behind these approaches is to reduce idle listening which is the time a node spends for listening to the medium even though no packets are being transmitted to it. An example of synchronized protocols is S-MAC [12]. In this kind of protocols, the idea is to synchronise the schedules of the neighboring nodes in order to specify when nodes are awake and asleep within a frame, reducing thus the time and energy wasted in idle listening. Asynchronous protocols, on the other hand, rely on *low power listening* (LPL) or preamble sampling. When a node has data to send, it transmits an extended preamble, and then sends the data packet. When a receiver wakes up, it listens to the channel. If a preamble is detected, it will

remains awake until the end of the preamble in order to determine whether the packet was destined to it or not. Then, if the receiver is not the packet's target it goes to sleep. There are two disadvantages of this approach. First, once a non-target receiver wakes up and detects a preamble, it would have to wait until the end of the preamble to determine that it is not the target and should go back to sleep, introducing thus, an *overhearing problem*. Second, target node has to wait the full period until the preamble is finished before the data/ack exchange, increasing the one hop delay. X-MAC is an asynchronous protocol that addresses the issues described before, reducing thus the energy consumption and delay. The idea is to use a *short preamble* which embeds information regarding the destination address within the preamble so the no-target nodes can quickly go back to sleep. In this way, overhearing problem is avoided. To address the second issue where target receiver has to wait until the end of the preamble, X-MAC proposes a *strobed preamble* which allows target receiver to interrupt the long preamble once it determines that it is the target receiver. When a node wakes up and receives a short preamble packet, it looks at the target node ID included in the packet. If the node is not the destination then it goes to sleep mode immediately. If the node is the destination node it remains awake for the subsequent data packet. Differently from traditional LPL where preamble is sent as a constant stream, X-MAC inserts small pauses between short preambles during the time within which the sender listens to the medium. During this time, receiver can send an *early acknowledgement* packet back to the sender. Once the sender receives an acknowledgement from the receiver it stops sending preambles and sends the data packet (Figure 1).

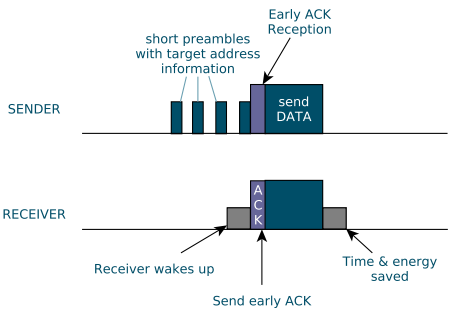


Figure 1: X-MAC short preamble approach.

B. RPL

RPL [10] is a Distance Vector IPv6 routing protocol for Low power and Lossy Networks (LLNs) that specifies how to build a Destination Oriented Directed Acyclic Graph (DODAG) using an objective function and a set of metrics/constraints. The objective function operates on a combination of metrics and constraints to compute the "best" path. There could be several objective functions in operation on the same node. For example, several DODAGs may be used with the objective to "Find paths with best ETX (Expected Transmission Count) values" or "Find the best path in terms of latency (metric) while avoiding battery-operated nodes (constraint)". In this paper we make use of the Contiki implementation of the RPL protocol so two objective functions are taken into account: the best path in terms of the Expected Transmission Count (ETX) and objective function 0, which is basically hop-count. Objective functions then dictate some rules to create the DODAG. A DODAG is a logical routing topology built over a physical network to meet a specific criteria and we can have multiples DODAGs active at the same time. A node then can participate and join one or more DODAGs (RPL instances). The DODAG is created by the root (configured by the system administrator) by an exchange of specific messages. Neighbors will receive the message and will make a decision whether to join the DODAG or not (based on the objective function, local policy, etc).
It is important to note here that, differently from our previous work, the fact of considering this routing protocol will introduce a dynamic routing not considered before. This is achieved when we consider the ETX objective function since its objective is to dynamically search for the best path in terms of the expected transmission count. Hence, the DODAG will oscillate between differents topologies during the execution of the protocol.
More information regarding RPL can be found in [10].

IV. Methodology

In this section we introduce the design and implementation of our Process Mining approach. Since this paper is an extension of previous work [3] and [4], we are not going to detail all the steps since we have already done this previously. However, we include the step of state's identification since we are considering a different MAC layer protocol. Besides, we extend the end to end delay computation module in order to consider dynamic routing.

A. Protocol Specification and States Identification

As we have mentioned in our previous works, all the states of the protocol should be taken from the corresponding protocol specification. In our case, we have developed the corresponding CX-MAC flow diagram from the specification and the implementation of the protocol (shown in Figure 2) to identify both states and transitions. When a packet arrives (PACKET ARRIVAL) to a sender node it would be enqueued (ENQUEUING) in the node buffer and will wait in the queue until previous existing packets in the queue are processed. Before waking up, the node go through the list of encounters to find if it has recorded an encounter with the receiver neighbor. If so the node would wait until the receiver is expected to be awake (switching to WAIT_NEIGHBOR) and then it wakes up. Otherwise, the sender will switch

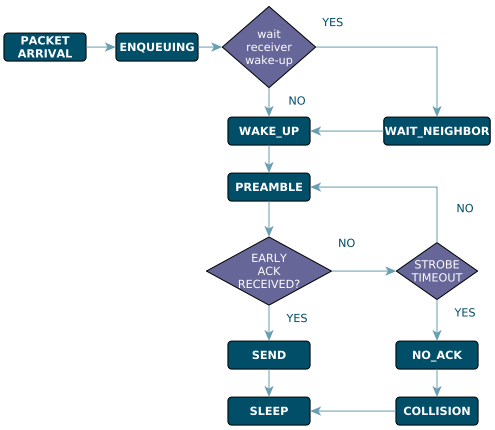


Figure 2: State's diagram of X-MAC protocol.

immediately to WAKE_UP state. Once the sender has waked up it will starts to send preambles (PREAMBLE). If between two preambles the sender receives an early ack it will send the packet (SEND) and then goes to sleep (SLEEP). Otherwise, it will continue sending preambles until it receives an early ack or until timeout for sending preambles is reached (NO_ACK) switching then to COLLISION state, which basically throws a notification to the upper layer and then go to sleep mode (SLEEP).

B. End to End Delay Computation

Finally, we implement the Performance Computation module which will compute the e2e delay between a source node n_s and the destination node n_d. In order to estimate this performance parameter we make use of the approach presented in [7] taking the obtained Markov chain as the input. We first start by computing the delay in one hop for each of the nodes extending then the analysis to estimate the e2e delay from source node n_s to the destination n_d (sink). In order to estimate the one hop delay we proceed as follows: the Markov chain will give us the information of the transitions between states of the protocol together with the corresponding transition probabilities. Then, it is possible to compute the *Probability Transition Matrix* P. This Markov chain has a unique initial state s_i and one or more final states $\{s_{f_1}, s_{f_2}, \cdots\}$. Since we are estimating the one-hop delay from the moment a packet arrive to the node until the packet is successfully sent, the only final state we consider for estimating the delay is the one associated to the reception of the acknowledge packet. Without loss generality, let's call this state s_f. We will call *sequence* to a path from state s_i to s_f. From P and the Laplace transform of the sojourn time[1] distribution e_{s_k} on each state s_k, we are able to compute the *Adjacency Matrix* A defined in [7] as follows:

$$A = \begin{pmatrix} 0 & p_{s_1 s_2} e_{s_1} & p_{s_1 s_3} e_{s_1} & \cdots & p_{s_1 s_f} e_{s_1} \\ 0 & 0 & p_{s_2 s_3} e_{s_2} & \cdots & p_{s_2 s_f} e_{s_1} \\ \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \quad (1)$$

where e_{s_k} represents the Laplace transform of the sojourn time distribution on state s_k and $p_{s_k s_l}$ the transition probability from state s_k to s_l taken from P. To find e_{s_k}, we compute the empirical average sojourn time γ_{s_k} obtained by analysing the traces of the protocol. Assuming that the sojourn time on state s_k follows a negative exponential distribution of parameter γ_{s_k}, the Laplace transform in frequency domain of the sojourn time distribution is

$$e_{s_k} = \frac{\gamma_{s_k}}{\gamma_{s_k} + s} \quad (2)$$

Once obtained A, we proceed to compute the vector $\vec{A}^r_{s_k,s_f}$ representing the delay distribution of all *sequences* of length r, $r = \{1, 2, 3, ...\}$, from state s_k to the final state s_f which is computed as follows:

$$\vec{A}^r_{s_k,s_f} = A \cdot \vec{A}^{r-1}_{s_k,s_f} \quad (3)$$

where $\vec{A}^1_{s_k,s_f}$ is the vector containing the delay distribution of all *sequences* of length = 1 from state s_k to the final one. This vector is a non null vector since there is always one or more states directly connected to the final state s_k. Being s_i the initial state representing the packet arrival event (initial state previously defined), $A^r_{s_i,s_f}$ represents the delay distribution in r steps (*sequences* of length = r) from s_i to s_f. Then, delay distribution of a node in the frequency domain can be computed as follows:

$$D_{f-dom}(s) = \sum_{r=1} A^r_{s_i,s_f} \quad (4)$$

The first order derivative of 4 evaluated in $s = 0$ will give us the average delay from s_i to s_f. The one hop delay distribution in time domain can be computed by means of the Inverse Laplace Transform (ILP)

$$D_{t-dom}(t) = ILP(D_{f-dom}(s)) \quad (5)$$

In order to compute the e2e delay from a given node n_s to the destination n_d (sink) we need to find the individually delay distribution in frecuency domain for each intermediate node n_k along the paths from n_s to n_d. In case the path is a serial path (Figure 3, top), the e2e delay distribution in frequency domain can be computed as the product of the individual one hop delay for each intermediate node n_k in the path from n_s to n_d

$$D_{e2e(f-dom)}(s) = \prod_{i=1}^{y} D^{(n_i)}_{f-dom}(s) \quad (6)$$

where n_n is the last hop node before reaching n_d and $D^{(n_k)}_{f-dom}(s)$ the one hop delay in frequency domain for

[1]Delay from the moment the system enters into a particular state until the moment the system changes to another one.

node n_k. Once again, derivating and evaluating equation 6 at $s = 0$ will give us the average end to end delay from source to destination. In case the path is a multi path (Figure 3, bottom) we can follow the idea from [7] for the parallel case (equation 2) computing the e2e delay distribution as follows:

$$D_{e2e(f-dom)}(s) = \prod_{i=1}^{j} D_{f-dom}^{(n_i)}(s) \cdot \left(\sum_{i=k}^{s} p_{j,i} \cdot D_{f-dom}^{(n_i)}(s) \right) \cdot \prod_{i=t}^{y} D_{f-dom}^{(n_i)}(s) \tag{7}$$

where p_{ki} is the probability for a packet arriving to n_k to be forwarded to node n_i. Finally, the whole e2e delay distribution in time domain can be computed as follows:

$$D_{e2e(t-dom)}(t) = ILP(D_{e2e(f-dom)}(s)) \tag{8}$$

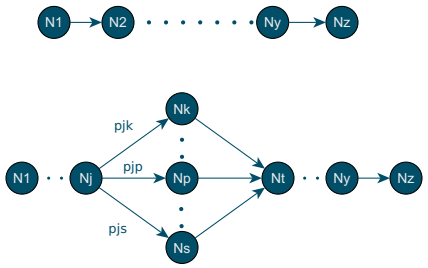


Figure 3: Serial (in tandem) and parallel structures.

V. Results & Discussions

A. Scenario Configuration

We have ran our experimentation in a large-scale infrastructure suitable for testing wireless sensor devices and heterogeneous communicating objects provided by IoT-LAB [6]. Different from [3] and [4], we have extended our experimentation to consider also a dynamic routing provided by the RPL protocol. The implementations of both X-MAC and RPL are the ones provided by Contiki OS. A total of seven nodes from the testbed were selected to carry out the experiments. Distances between them are shown in Table I. Here, node 145 was configured as the sink node while the others generate packets with a given inter-arrival rate and send them to the sink. The scenarios were divided in two groups, the first group uses the RPL objective function 0 (OF0) while the second the RPL objective function ETX as the routing metric For each of these groups, a set of four scenarios with different arrival rates $\lambda = 0.5p/s$, $\lambda = 1p/s$, $\lambda = 2p/s$, $\lambda = 4p/s$ were defined. The packet size was defined as 42 bytes (25 bytes of payload + 17 bytes of header) and the queue length on each node was set to 8 packets. Power transmission on each node was set to -3dBm.

	145	205	236	166	45	14	54
145	*	4.16	9.97	10.03	12.53	9.16	4.5
205	-	*	7.89	9.39	13.81	11.47	7.91
236	-	-	*	3.51	9.82	10.24	10.46
166	-	-	-	*	6.32	7.29	8.93
45	-	-	-	-	*	4.30	9.09
14	-	-	-	-	-	*	5.1
54	-	-	-	-	-	-	*

Table I: Distances (meters) between selected nodes.

The simulation time for $\lambda = 0.5$ and $\lambda = 1$ was set to 16 minutes while for $\lambda = 2$ and $\lambda = 4$ was 10 minutes. It is important to note that we have specifically selected a high traffic load scenario ($\lambda = 4$) in order to test the performance of our approach under a saturated scenario. For the sake of simplicity and in order to avoid having a huge Markov model, we do not consider retransmissions due to collisions in our experimentations. However, the approach is still valid if we consider also retransmissions.

B. Experimentation Results

1) Markov Chain: In Figure 4, the Markov chain for node 166 is presented. However, because of the lack of space, we show the results for only one node. Except for leaf nodes where packets arrive from the application layer, for internal tree nodes packets might arrive either from the node's application layer, from some of the nodes in the neighborhood, or both. It is important to note that a Markov chain like this is obtained for each of the nodes belonging to the network and would be helpful for estimating the delay in one hop for each node and from them the whole end to end delay. Two particular states are presented in this Markov chain: BUFFER FULL and PACKET DROPPED representing those packets that were dropped due to the fact that the buffer node was full and packets that were dropped after the preambule's period, respectively. Regarding the transition probability from state PREAMBULE_24 to ACK_RECEIVED, we can see that its value is equals to one. This is due to the fact that this Markov chain represents the behavior of the protocol for those packets that effectively reached the next destination. Packets that were dropped at the end of the preambule's period will, obviously, have an impact over the local delay. Markov chains for each scenario can be found in [2]. Then, this Markov chain represents the behavior of the CX_MAX protocol running locally in the node. On the other hand, RPL protocol works over the MAC layer and thus its scope covers the totality of the network (by determining the best path from one node to the sink, assigning node parents, etc). Hence, and in order to estimate the one hop delay in one node, it is enough to have a Markov chain modelling the behavior of the underlying MAC protocol.

2) One-hop and End to End delay: As we mentioned before, experimentations were done by defining two sets of scenarios, the first considering the OF0 RPL objective

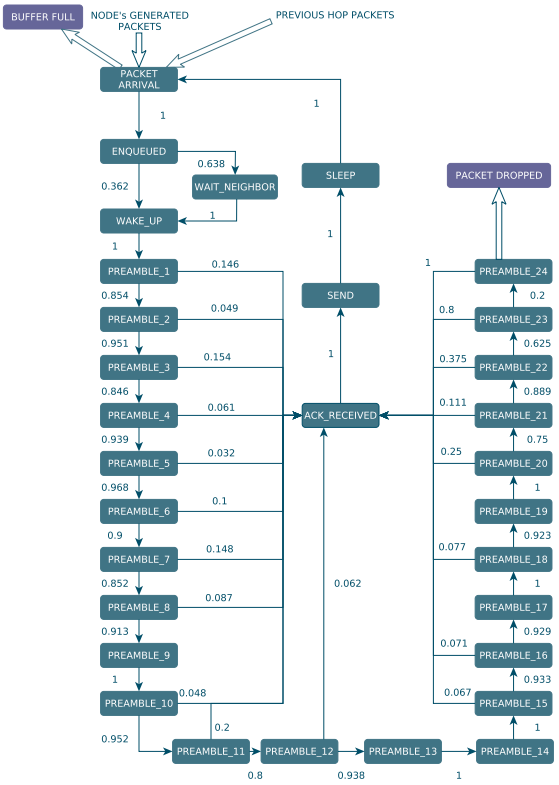
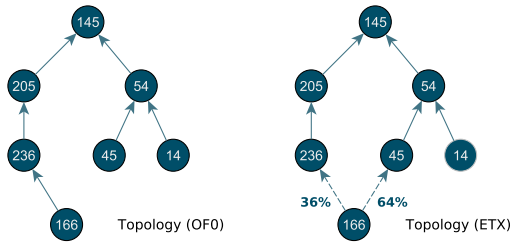


Figure 4: Markov chain model for node 166 ($\lambda = 4$).

function and the second one considering the ETX. Then, for each subset we executed the experimentation considering four different arrival rates values. Figures (5a) and (5b) show the routing graphs (DODAGs) generated by the RPL protocol for both objective functions during the execution of the experimentation. In the case of the ETX objective function scenario, for high traffic load ($\lambda = 4$), the parent selection mechanism of node 166 fluctuates between nodes 45 and 236 in such a way that, at the end of the execution, 64% of the generated packets were sent to the sink through node 45 while the rest 36% of the packets were sent through node 236. Tables II and III present, for each node, the average one-hop and e2e delay empirically measured together with the theoretical average delay estimation for both one-hop and e2e obtained by means of the methodology described in the previous section. The empirical average delays were obtained by measuring the delay for each generated packet. In order to be able to make a comparison in terms of delay between the two objective functions, we have computed the global average delays. For each scenario (objective function/λ) we compute the average one-hop delay among all nodes in the network. The same method is applied to the e2e average delay for each scenario. A summary of these results is shown in Table IV. Then, we were able to compare both metrics by looking at the global one-hop and e2e delay results. Figure 6 shows the probability density function of the e2e delay from node 166 to the sink (145) for both RPL objective functions. The e2e delay distribution was obtained by means of the analysis done in section IV-B. We only consider the high traffic load scenario ($\lambda = 4$) to plot since it is the scenario where results for both objective functions are more significantly different. Finally, Table V shows the percentage of the Packet Reception Rate (PRR) for each objective function.



(a) OF0 and ETX topology (b) ETX topology ($\lambda = 4$).

OF0	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 0.5$	205	0,1282	0,1282	205	0,1293	0,1293
	14	0,1417	0,2178	14	0,1430	0,2626
	54	0,1192	0,1192	54	0,1197	0,1197
	45	0,1561	0,2768	45	0,1550	0,2747
	166	0,1384	0,4272	166	0,1371	0,4275
	236	0,1618	0,2747	236	0,1611	0,2904
	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 1$	205	0,1193	0,1193	205	0,1208	0,1208
	14	0,1634	0,2625	14	0,1597	0,2851
	54	0,1251	0,1251	54	0,1254	0,1254
	45	0,1394	0,2396	45	0,1418	0,2672
	166	0,1920	0,4505	166	0,1822	0,4657
	236	0,1651	0,2663	236	0,1627	0,2835
	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 2$	205	0,1278	0,1278	205	0,1278	0,1278
	14	0,1851	0,2949	14	0,1806	0,3066
	54	0,1277	0,1277	54	0,1260	0,1260
	45	0,1706	0,2869	45	0,1677	0,2937
	166	0,1503	0,3917	166	0,1406	0,4366
	236	0,1720	0,2898	236	0,1681	0,2959
	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 4$	205	0,1428	0,1428	205	0,1432	0,1432
	14	0,2127	0,3948	14	0,1970	0,3668
	54	0,1635	0,1635	54	0,1697	0,1697
	45	0,2027	0,3737	45	0,2052	0,3749
	166	0,2055	0,4835	166	0,2055	0,5359
	236	0,1915	0,3553	236	0,1870	0,3303

Table II: Empirical and theoretical e2e average delay (OF0).

C. Discussions

Figures (5a) and (5b) show the DODAGs created by the RPL routing protocol during the experimentation. For $\lambda = 0.5$, $\lambda = 1$ and $\lambda = 2$, both metrics share the same topology (5a). On the other hand, as traffic rate increases ($\lambda = 4$) and due also to the fact of having a buffer size of eights packets on each node, the quality of the path in

ETX	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 0.5$	205	0,1123	0,1123	205	0,1111	0,1111
	14	0,1559	0,2655	14	0,1469	0,2656
	54	0,1202	0,1202	54	0,1187	0,1187
	45	0,1656	0,2831	45	0,1652	0,2839
	166	0,1877	0,4503	166	0,1865	0,4585
	236	0,1604	0,2670	236	0,1609	0,2720
	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 1$	205	0,1173	0,1173	205	0,1157	0,1157
	14	0,1580	0,2590	14	0,1592	0,2837
	54	0,1269	0,1269	54	0,1245	0,1245
	45	0,1536	0,2798	45	0,1549	0,2793
	166	0,1690	0,4158	166	0,1751	0,4449
	236	0,1552	0,2571	236	0,1542	0,2698
	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 2$	205	0,1283	0,1283	205	0,1252	0,1252
	14	0,1886	0,3034	14	0,1821	0,3120
	54	0,1327	0,1327	54	0,1299	0,1299
	45	0,1754	0,2926	45	0,1744	0,3044
	166	0,1639	0,4119	166	0,1518	0,4554
	236	0,1790	0,2939	236	0,1780	0,3033
	Empirical Av. Delay (sec)			Theoretical Av. Delay (sec)		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 4$	205	0,1305	0,1305	205	0,1296	0,1296
	14	0,1989	0,3415	14	0,1847	0,3245
	54	0,1423	0,1423	54	0,1398	0,1398
	45	0,2009	0,3367	45	0,1921	0,3320
	166	0,1143	0,4476	166	0,1144	0,4511
	236	0,2240	0,3898	236	0,2155	0,3451

Table III: Empirical and theoretical e2e average delay (ETX).

OF0	Emp. Global Av. Delay (sec)		Theo. Global Av. Delay (sec)	
λ	1-hop	e2e	1-hop	e2e
0.5	0,1409	0,2406	0,1408	0,2507
1	0,1507	0,2439	0,1488	0,2580
2	0,1556	0,2531	0,1518	0,2644
4	0,1864	0,3189	0,1846	0,3202
ETX	Emp. Global Av. Delay (sec)		Theo. Global Av. Delay (sec)	
λ	1-hop	e2e	1-hop	e2e
0.5	0,1503	0,2497	0,1482	0,2516
1	0,1467	0,2427	0,1472	0,2530
2	0,1613	0,2605	0,1569	0,2717
4	0,1685	0,2981	0,1627	0,2870

Table IV: Empirical and theoretical global average delay for each λ (OF0 & ETX).

	OF0	ETX
λ	PRR (%)	PRR (%)
0.5	97	98
1	96	95
2	33	35
4	2.8	9.9

Table V: Packet reception rate (PRR) for OF0 and ETX.

terms of expected transmission count between node 166 and the initial assigned parent (45) decreases. Hence, RPL parent selection mechanism for node 166 starts to oscillate between nodes 236 and 45 and this phenomenon is repeated time and time again throughout the experiment. This gives an alternative topology (5b) for the case of high traffic load. Tables II and III shows the empirical and theoretical delay for each node on each scenario. When comparing empirical vs theoretical delays we can see that, for the 1-hop case,

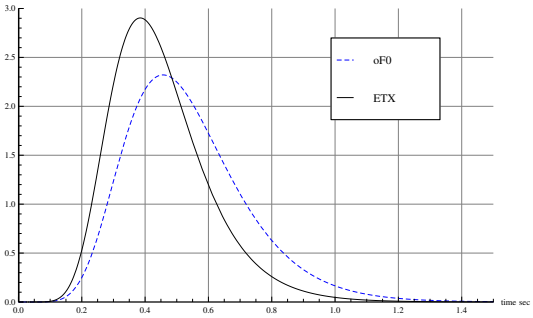


Figure 6: E2e delay distribution: OF0 and ETX for node 166 (λ = 4p/s)

empirical and theoretical delays are almost the same with an average difference between them of $\approx$ 2 milliseconds. For the case of e2e delay estimation, we can see that, for certain nodes, theoretical estimation differs a little bit from empirical measures. However, the average difference is $\approx$ 10 milliseconds and thus, the proposed approach described in the last section remains an acceptable way for estimating the e2e delay in a WSN. Table IV show results in terms of the global average delay for each of the scenarios. The first conclusion we can deduce from this result is that, as traffic load increases, the average delay also increases. This is due to the fact that the network congestion also increases and since the buffer length is limited (8 packets) then the time taken for each packet to be dispatched also increases and thus the e2e delay from the source node to the destination (sink). Concerning the comparison between both metrics, we can see that for low and moderate traffic load ($\lambda = 0.5$, $\lambda = 1$ and $\lambda = 2$), the global average delays are almost the same. On the other hand, when considering a high traffic load ($\lambda = 4$), the global average delay is significantly lower for the ETX case. This is a consequence of the RPL ETX strategy which builds an optimal path in terms of link quality and avoids sending packets over a congested link. The alternative path is supposed to be better in terms of link quality and congestion and thus it should improve network performance in terms of e2e delay. This can be clearly observed from the obtained delays of node 166. We can see from Tables II and III that for low and moderate traffic load scenarios, the e2e delay of node 166 does not considerably differ between the two metrics while for the high traffic load scenario, the obtained e2e delay, by means of ETX, is significantly lower than the one obtained by OF0. This can also be seen in Figure 6 where we show the probability density distribution of the e2e delay from node 166 to the sink (145) for both RPL objective functions. As shown in the plot, the whole e2e delay is significantly lower when considering ETX instead of OF0 objective function. Finally, Table V shows the percentage of the Packet Reception Rate (PRR) for each objective function. Intuitively, we can expect the PRR for the ETX metric to be greater than the PRR for

the OF0 since ETX looks for the best path from source to destination in terms of the expected transmission count. This issue can be confirmed from the results, specially, for the high traffic load scenario where PRR is improved ($\approx 7\%$) when considering the ETX metric. An important point to remark regarding the Markov chains is that they depend on input parameters, in particular, on the packet arrival rate. We have then though to find a way of generalising these Markov chains in such a way that they do not depend on the arrival rate. We have then started to work in this direction developing an approach based on non-linear regression in order to find mathematical expressions for both transition's probabilities and state's sojourn delay. We still working on this and results so far are encouraging.

VI. Conclusions & Future Work

In this paper, we have presented a new approach allowing to obtain the e2e delay expression in WSNs running both dynamic MAC and routing protocols. Markov chain models were obtained for a network running X-MAC on each node and also considering the RPL routing protocol. It is a significant extension of our previous work in [3] and [4]. The contributions of our work can be enumerated as follows:

- We were able to find an analytical expression of the e2e delay distribution in a WSN for a real scenario considering, not only the underlying MAC protocol, but also a routing protocol.
- By means of our model we were able to compare two routing strategies in terms of end to end delay in a multi-hop transmission scenario.
- Our approach compute the e2e delay from individual one-hop delay distribution taken from individual Markov chains modelling the behavior of local underlying MAC protocol. Hence, we can conclude that our approach is suitable for estimating the e2e delay independently of the chosen routing protocol.

Our future work concerns the generalisation of the Markov chains from sample arrival rates, thus by means of non-linear regression in order to obtain an approximate general Markov chain that does not depends on a specific arrival rate.

References

[1] E. Ancillotti, R. Bruno, and M. Conti. Reliable data delivery with the ietf routing protocol for low-power and lossy networks. *Industrial Informatics, IEEE Transactions on*, 10(3):1864–1877, Aug 2014.

[2] Francois Despaux, Ye-Qiong Song, and Abdelkader Lahmadi. Markov Chain Results. urlhttp://qoswsanquasimod.gforge.inria.fr/.

[3] Francois Despaux, Ye-Qiong Song, and Abdelkader Lahmadi. Towards performance analysis of wireless sensor networks using process mining techniques. In *ISCC 2014*.

[4] Francois Despaux, Ye-Qiong Song, and Abdelkader Lahmadi. Modelling and performance analysis of wireless sensor networks using process mining techniques: Contikimac use case. In *IEEE DCOSS 2014*, pages 225–232, 2014.

[5] Messaoud Doudou, Djamel Djenouri, and Nadjib Badache. Survey on latency issues of asynchronous MAC protocols in delay-sensitive wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 15(2):528–550, 2013.

[6] Olivier Fambon, Eric Fleury, Gaetan Harter, Roger Pissard-Gibollet, and Frederic Saint-Marcel. FIT IoT-LAB tutorial: hands-on practice with a very large scale testbed tool for the Internet of Things, 2014.

[7] Wenbo He, Xue Liu, Long Zheng, and Hao Yang. Reliability calculus: A theoretical framework to analyze communication reliability. ICDCS '10, pages 159–168, Washington, DC, USA, 2010. IEEE Computer Society.

[8] Jelena Misic and Vojislav Misic. *Wireless Personal Area Networks: Performance, Interconnection, and Security with IEEE 802.15.4*. Wiley Publishing, 2008.

[9] P. Park, P. Di Marco, C. Fischione, and K. H. Johansson. Delay analysis of slotted IEEE 802.15. 4 with a finite retry limit and unsaturated traffic. In *IEEE Global Communications Conference*, pages 1–8, 2009.

[10] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550 (Proposed Standard), 2012.

[11] Ou Yang and Wendi Heinzelman. Modeling and performance analysis for duty-cycled MAC protocols with applications to s-MAC and x-MAC. *IEEE Transactions on Mobile Computing*, 11(6):905–921, 2012.

[12] Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *INFOCOM 2002.*, volume 3, pages 1567–1576, 2002.