Towards performance analysis of wireless sensor networks using Process Mining Techniques

François Despaux, Ye-Qiong Song, Abdelkader Lahmadi

Towards Performance Analysis of Wireless Sensor Networks Using Process Mining Techniques

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*—Performance analysis of wireless sensor networks is a difficult task because of the high dynamic of networks and the use of duty-cycled MAC protocols. Markov-based modelling is an interesting approach to deal with this problem. However, existing Markov-based analytic models, being MAC protocol-centric rather than network-centric, work under strong assumptions and do not allow to encompassing important network parameters like radio channel fading and capture effect, or actual implementation optimizations (not always specified in the protocol description). In this paper we propose a novel approach to obtain a Markov chain model for networks running different MAC protocols by means of Process Mining Techniques. We present the main aspects of our approach together with the results obtained for the standard IEEE 802.15.4. The obtained Markov model can be used to evaluate various performance parameters. The approach can also be extended to a wider range of protocols.**

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

I. Introduction

Understanding the behavior and limitation of the wireless sensor networks 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 and further take decisions for improving the network performance. A lot of research work has been done to model the network through different methods, including analytical modelling and simulation based analysis. Due to its high dynamic nature, wireless sensor networks 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 in order to simplify the analysis and thus they are not enough accurate for estimating the performance parameters. Let's just take the example of the widely spread standard IEEE 802.15.4 MAC protocol [1]. In [9], authors present a Markovian model with which a set of performance results were obtained. However, a queue of size one was considered on each node, which does not represent the reality. 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 nodes were modeled as a $M/G/1/K$ queue system. However, the model lacks of a realistic radio channel model and capture effect model. Besides, the extension of the analysis to more complex multi-hop networks is not feasible because the input flow to intermediate nodes is no longuer Markovian but a general process (G/G/1/K) difficult to analyse.

Many other analytic models have been proposed in the literature in order to model the behavior of other duty cycled wireless sensor network MAC protocols such as S-MAC, X-MAC. Those models catch some main features of the protocols to give asymptotic performance trends in function of traffic load, allowing generally a qualitative rather than quantitative comparison of protocols. However, when one is interested by evaluating the performance of a network running a protocol, but not a protocol itself, it is still very difficult, to not say impossible, to apply those existing models. One of the reasons is that most of those models don't include neither network-related parameters (e.g. actual channel model, capture effect) nor actual implementation details (OS and implementation limits and optimizations). Otherwise the model would be too complex to be analytically resolvable. This difficulty is more stringent as soon as multi-hop network is concerned, since the input flow to the forwarders is generally unknown and not necessarily Markovian. There exists only few work dealing with multi-hop network performance analysis.

In this paper we propose a novel approach for modelling the network behavior. Our approach combines the measurement-based and analytic approaches. Differently from the existing performance measurement methods which directly focus on the performance metrics, we first instrument the protocol code and record the protocol execution trace on network nodes in a log file (rather than sniffer's traffic trace) to capture both implementation details and network physical parameters. Considering that the approach requires a protocol execution to generate the log files one can ask why do not directly measure the performance from the execution

[0]This work was partially supported by Quasimodo project under No. ANR 2010 INTB 0206 01 in France and No. NSFC 61061130563 in China

output. The answer to this question is that we are interested in finding a Markov model for modelling the protocol behavior that will allows us to estimate the probability distribution of the e2e delay. A measuring approach will give us the average e2e delay and would require a lot of executions and samples in order to estimate the probability distribution. In the second step, we use the process mining approach to extract an Markov chain that more accurately models the network behavior. This Markov model can then be used to further evaluating the performance parameters such as delays of the network. Of course the extracted Markov chain model is traffic dependent. Nevertheless, a useful practice may be to generate traces for several scenarios with different traffic patterns (e.g. light, medium and heavy traffic) and network conditions, allowing us to extract more general conclusions of the protocol behavior. Finally for computing the end-to-end delays in a large scale multi-hop network, this approach allows to bypass the difficulty of modelling the input flows of the forwarders (generally not Poisson arrivals), by directly using the Markov chain of those nodes.

The main contributions of this work are summarized as follows.

- We show how to make novel use of Process Mining technique to extract Markov chains from protocol execution traces.
- Our approach is an alternative way for modelling the network that encompass phenomenons not taken into account by existing theoretical models.
- Performance metrics can be computed from the extracted Markov chain model, specially the end to end delay in a multi-hop tree scenario.

In this work we focus on the network MAC layer for obtaining a comprehensive Markov model of the widely spreaded IEEE 802.15.4 standard MAC protocol. Our approach was also extended to model the ContikiMAC protocol where we computed the e2e delay distribution in a basic multi-hop tree topology.

The reminder of this paper is organized as follows. Section II presents main related existing work on analytic modelling of duty-cycled MAC protocols. Section III gives a background of the standard IEEE 802.15.4 protocol, 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 discussions and conclusions are presented in Section VI and VII respectively.

II. Related Work

In this section we focus on the review of the main analytic models for the standard 802.15.4 as well as some work related to our approach. Most of the proposed solutions for IEEE 802.15.4 are based on Bianchi's Markov model [2], initially developed for IEEE 802.11 standard. This model has been extended for modelling the IEEE 802.15.4 MAC protocol under different assumptions. In [9], authors proposed a Markov chain approach for modelling the slotted version of the CSMA/CA mechanism in IEEE 802.15.4 MAC protocol and have given performance results in terms of service time and delay for successful packet transmission. The probability distribution of the packet delay is derived from this model. A limitation of this model is that the queue capacity on each node is fixed to one packet and duty cycle is not considered. Therefore, this model is not suitable for modelling a real wireless sensor network scenario. Misic et al. [8] proposed a Markov chain model for the standard IEEE 802.15.4 MAC protocol considering a $M/G/1/K$ system 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. Even considering a $M/G/1/K$ queue system for the first node, taking into account that the output distribution of a $M/G/1/K$ is not 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. All the above mentioned works only deals with single hop case, so they cannot be readily used for evaluating multi-hop networks. [15] is one of the rare work dealing with multi-hop network. A more general framework is proposed for including both channel, MAC and routing characteristics in the analysis. By considering the TinyOS default CSMA/CA MAC protocol (similar to IEEE802.15.4), each node is modeled by a Geom/PH/1/M queue. The e2e delay distribution is obtained and compared to both simulations and measurements. This is the most achieved work. Its extension for dealing with dynamic duty-cycled MAC protocol is however not obvious. Authors in [14], propose a system that can automatically infer a protocol state machine from real-world traces. However, their approach is based on network traces where normally no information regarding the underlying MAC protocol behavior is present. Besides, the suitability of this approach for estimating performance parameters such as e2e delay is not clear since the output of the system is a state machine where no information concerning the sejourn time on each state is available. Finally, authors in [6] developed a theoretical framework to estimate the end-to-end delay in a networked system using frequency-domain modelling and analysis where they 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. Backgrounds

A. Overview of IEEE 802.15.4 Standard

The IEEE 802.15.4 standard defines the PHY and MAC sublayer specifications for low-rate WPANs (LR-WPANs). Like the IEEE 802.11 protocol, the standard makes use of CSMA/CA as the channel access protocol and it also brings support for contention-free and contention-based periods. Two operational modes are supported, *beacon enabled* and *non beacon-enabled*. In this paper, we focus our attention in the *beacon enabled* mode of the protocol. In this mode, a superframe structure is proposed in order to manage the communication between devices. The superframe format is defined by the PAN coordinator and is sent to the other devices within each beacon frame. As seen in Figure 1, the

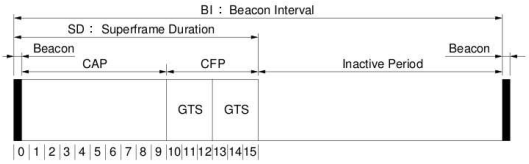

Figure 1: Superframe Structure.

superframe structure defines an active and an inactive period. The length of these periods is defined by two parameters: *macBeaconOrder(BO)* and *macSuperframeOrder(SO)*. The former determines the interval at which the coordinator must transmit beacon frames. The second parameter describes the length of the active portion of the superframe. Finally, BO and SO must satisfy the constraint $0 <= SO <= BO <= 14$. The duty cycle is the ratio of the length of an active period *SO* and the length of a cycle time *BO*, and is calculated as $\left(\frac{1}{2}\right)^{BO-SO}$. In this way, by handling both *SO* and *BO* we can get different duty cycle configuration. Figure 2 shows the flow diagram for the slotted version of the CSMA/CA mechanism in the standard IEEE 802.15.4 mac protocol. Here, *NB* represents the number of times the CSMA/CA algorithm will enter in backoff while attempting the access to the current channel, *CW* represents the number of times the CSMA/CA will check the channel availability before starting transmission and *BE* represents the backoff exponent. Each time the channel is found busy *BE* is incremented by 1 until it reach the maximum possible value *aMaxBE* which is a constant defined in the standard it has a default value equal to 5.

B. Process Mining

Process mining has been widely applied in lots of fields and it is an analysis method to construct models automatically through analysing the event logs. It can be considered a branch of data mining. Traditional data mining methods aims at forecasting system behaviors while Process Mining at constructing whole process models. Mining algorithms is a key aspect in Process Mining. Many algorithms has

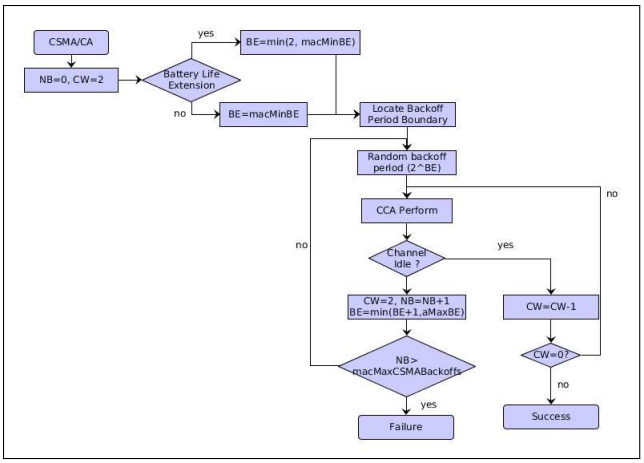

Figure 2: Slotted CSMA/CA flow diagram.

been proposed in order to construct process model from event log files. Van der Aalst [11] propose the α and β algorithms for discovering a workflow model based on Petri nets and developed the ProM mining tool [12]. Authors in [7] propose a method with derivation and statistics which uses Stochastic Task Graphs as the intermediate to obtain a workflow models. In [13], authors propose a sequence clustering algorithm for processes with high diversity of behavior. The algorithm consists in dividing the log into clusters in order to analyse reduced sets of cases and find a Markov chain model for constructing the process model. The algorithm was implemented as a plugin in ProM. In this work, we make use of this algorithm in order to obtain a Markov chain for modelling the network behavior. Given a set of clusters c_k, the algorithm starts by randomly initializing the state transition probabilities. The second step is to assign each sequence to the cluster that can produce the higher probability (see [13] for the probability expression). New transition probabilities are computed for each cluster and then the algorithm repeats the assignment of sequences to cluster until the cluster models do not change. In our case we consider only cluster and one sequence containing all the states and transitions during the protocol execution so the algorithm is reduced to find the frequences of transitions between states within the whole sequence.

IV. Methodology

In this section we introduce the design and implementation of our Process Mining approach. The Process Mining tool used in this work is the ProM data mining tool version 5.2. As we said before, we focus our attention on the network MAC layer so the analysis done here studies the underlying mac protocol of the network. However, and since the Process Mining takes in consideration the network traces, the approach can be easily extended to cover the whole network behavior. In Figure 3 we can see a flow diagram showing the steps to follow in order to obtain the Markov chain model from the protocol traces. In the next subsections

we describe each of the components of the flow diagram.

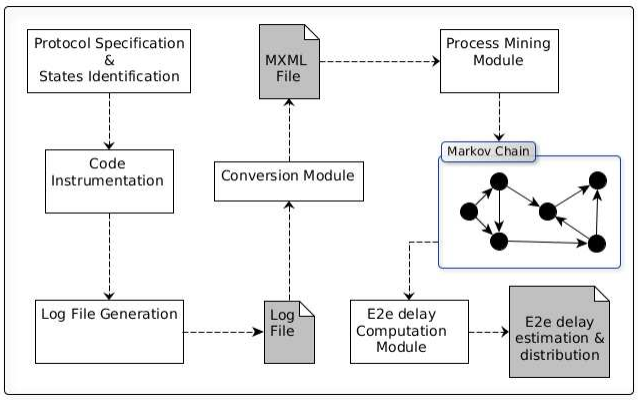

Figure 3: Step by step flow diagram.

A. Protocol Specification and States Identification

The first and one of the most important step in this process is the protocol specification and states identification. Each protocol has a specification from where we can obtain the main aspects of it. Usually, a flow diagram showing the main states and transitions between them is provided. Based on the specification of this protocol we can identify the main states and transition between them. All these states should be taken from the corresponding protocol specification. In our case, we make use of the IEEE 802.15.4 mac protocol flow diagram shown in Figure 2 to identify both states and transitions.

B. Code Instrumentation & Log File Generation

Once we have identified each state and transitions in the protocol, the next step is to generate the log files in such a way that all states and transitions previously identified are present in the protocol execution output (log file). In order to generate the log file, it would be necessary to identify each state in the protocol implementation so that each state will appear in the protocol execution and therefore in the generated log file. In this way, the procedure consists in printing a line each time a change from one state to another is found during the execution. In our case, we make use of TKN154 [5] protocol implementation. TKN154 is a platform independent IEEE 802.15.4-2006 MAC implementation for the 2.1 release of the TinyOS execution environment. Since the TKN154 code is written in C, it is enough to add a *printf* command in the code whenever a new state is reached by the execution. In this way, we will obtain a trace log from the execution where each state should be present and transitions are represented by the previous and next state of the current state, that is to say, if state X is followed by state Z and preceded by state Y in the log file, then we have both Y $\rightarrow$ X and X $\rightarrow$ Z transitions.

C. Model Extraction & Performance Computation

1) Conversion Module: The purpose of this module is to translate the network event log file into a readable MXML file that would be consumed by the Process Mining tool. Therefore, it would receive the event log file as the input and after processing it sequentially, it will translate it in MXML format. This format follows a specified schema definition, which means the log does not consist of random and disorganized information. Rather it contains all the elements needed by the plug-ins at a known location.

2) Events Mining and Model Extraction: The environment in which this work is based is the ProM Process Mining tool version 5.2. The output of the parsing step done by the *Conversion* module provides us the input for the ProM tool which is the event log file in MXML format. We make use of the sequence clustering technique in order to find a Markov chain model from the event log file since it can be considered as a sequence of states. An implementation of this algorithm is provided by ProM. Then, we are able to find the Markov chain model by loading the MXML file generated by the *Conversion* module and applying it to the the sequence clustering algorithm.

3) Performance Computation Module: Finally, we implement the Performance Computation module which will compute the end to end delay between the source and the destination node. In order to estimate this performance parameter, we make use of the approach presented in [6] taking the obtained Markov chain as the input. The Markov chain will give us the information on the transitions between states of the protocol together with the corresponding transition probabilities. Then, it is possible to compute the *Probability Transition Matrix* P. From P and the Laplace transform of the sejourn time distribution on each state we are able to compute the *Adjacency Matrix* A defined in [6]. Once obtained A, we proceed to compute the vector $\vec{A}^r_{i,d}$ representing the delay distribution of all connected paths of length r, $r = \{1, 2, 3, ...\}$, from state i to destination d. Then, the e2e delay distribution can be found from the set of vectors $\vec{A}^r_{i,d}$ for $r = \{1, 2, 3, ...\}$, as we will see in next section.

V. Experiments & Results

A. Scenario configuration

In order to carry out the experimentation, we have set a testbed with TelosB motes, TinyOS as the underlying operating system and the TKN154 IEEE 802.15.4 mac implementation. We consider a tree topology as shown in Figure 4 where devices (Dx) periodically send unicast packets to a specific router node (Rx). Once receiving a packet, the router forwards it to the coordinator (C). We set three scenarios by varying the Poisson arrival rate to each device: 1, 5 and 10 packets per second. The packet size is set to 32 bytes (21 bytes of payload + 11 bytes of header

and trailer) and the queue length on each node was set to four packets. The mac protocol parameters are the ones by default with a specific duty cycle of 50% ($BO = 6$ and $SO = 5$). The transmission power of each node is 0dBm and the distance between device and router, as well as the one between router and coordinator was set to 1 meter. We have obtained the protocol traces by executing the experiments during five minutes.

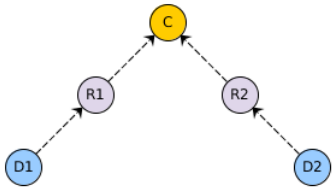

Figure 4: TelosB scenario.

B. Resulting Markov chain

In order to obtain the Markov chain model, we apply the methodology described in previous section to the concerned nodes in the network. Since we are interested in measuring the end to end delay from a given device to the coordinator, we focus our attention on a particular branch of the topology, for instance, the D1-R1-C branch. Then, we proceed to obtain both device and router logs and by applying the procedure described before, we get the corresponding Markov chain model for each node in the path. Due to the lack of space, we only present the Markov chain result for the device (Figure 5). A similar result with some differences in states and transition probabilities was obtained for the router node. Two subscripts were added to each state in order to represent the number of collisions and number of times the channel was found busy. Then, each state in the Markov chain has the *STATE_k_l* format where k represents the current number of collisions and l the number of times the channel was found busy. For instance, being in state *CCA2_0_0*, if a collision occurs then we increment the k variable and the Markov chain moves to the *START BACKOFF_1_0* to retry the channel assessment.

C. End to end delay estimation

Now it is time to estimate the e2e delay in two hops from devices to the coordinator. As we explained before, based on the obtained Markov chain model and the transition probabilities between each state we are able to compute the *Probability Transition Matrix* P. From P and the estimation of the sejourn time distribution e_i on each state of the Markov chain, we compute the *Adjacency Matrix* A as follows

$$A = \begin{pmatrix} 0 & p_{12}e_1 & 0 & \cdots & 0 \\ 0 & 0 & p_{23}e_2 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \quad (1)$$

where e_i represents the Laplace transform of the sejourn time distribution of the state i and p_{ij} the transition probability from state i to j taken from P. To find e_i, we compute the empirical average sejourn time γ_i obtained by analysing the traces of the protocol. Then, we make the assumption that the sejourn time on state i follows an exponential distribution of parameter γ_i. Therefore, the Laplace transform of the sejourn time distribution is

$$e_i = \frac{\gamma_i}{\gamma_i + s} \quad (2)$$

Once obtained A, we proceed to compute the vector $\vec{A}^r_{i,d}$ representing the delay distribution of all connected paths of length r, $r = \{1, 2, 3, ...\}$, from state i to destination d (ACK RECEIVED) which is computed as follows:

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

where $\vec{A}^1_{i,d}$ is the vector containing the delay distribution in one-hop from state i to the destination. This vector is a non null vector since there is always one or more states directly connected to the destination state. Being s the source state (ENQUEUING), when we look at $A^r_{s,d}$ we find the delay distribution in r hops from source to destination. Then, the whole delay distribution in the frequency domain can be computed as follows:

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

The first order derivative of 4 evaluated in $s = 0$ will give us the average delay $\bar{d}$ of the Markov chain. This procedure is applied for both router and device Markov chains obtaining $\bar{d}_{device}$ and $\bar{d}_{router}$. Then, the average e2e delay from device to coordinator is the sum of these values

$$\bar{D}_{e2e} = \bar{d}_{device} + \bar{d}_{router} \quad (5)$$

The whole e2e delay distribution in frequency domain $D_{e2e(f-dom)}$ is the product of (4) for both device and router. Finally, the whole e2e delay distribution in time domain can be computed by means of the Inverse Laplace Transform applied to $D_{e2e(f-dom)}$. Figure 6 shows the e2e delay computed by the Performance Computation Module while Table I shows the comparision between the measured e2e delay and the computed one for the three defined scenarios.

	Empirical Av. Delay (sec)			Computed Av. Delay (sec)		
λ	Device	Router	e2e	Device	Router	e2e
1 p/sec	0.1577	0.4481	0.605	0.1578	0.4483	0.606
5 p/sec	0.22	0.498	0.718	0.22	0.498	0.719
10 p/sec	0.345	0.484	0.83	0.345	0.484	0.83

Table I: Empirical and computed e2e average delay.

From Figure 6 we can see that, as the arrival rate increases, the e2e delay also increases. This is due to the fact that for low traffic scenarios ($\lambda = 1$ p/s) the queue is almost

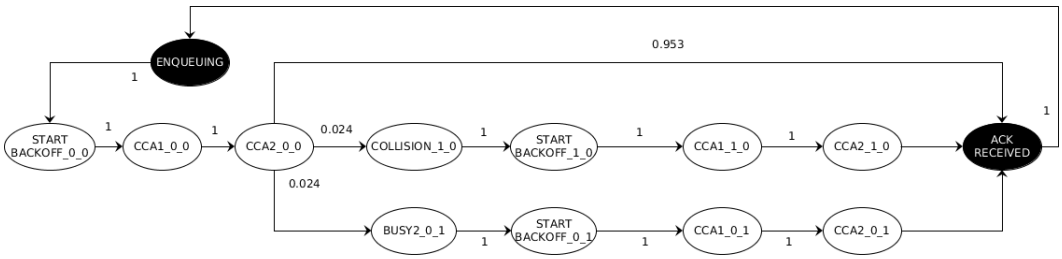

Figure 5: Obtained Markov chain of a single device.

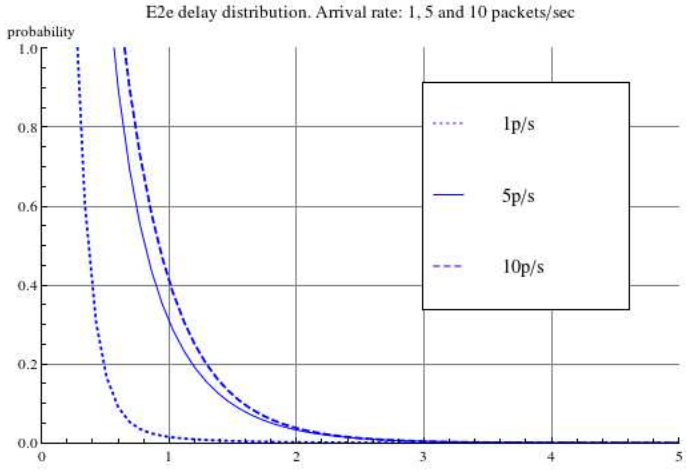

Figure 6: Probability distribution function of e2e delay for packet arrival rate $\lambda = 1, 5, 10$ p/s.

λ	% Packet dropped	
	Device	Router
1 p/sec	0	0
5 p/sec	12	0
10 p/sec	50	0

Table II: Buffer drop rate.

empty all the time and then queuing delay is minimum. We can see from Table II that for the low traffic scenario none of the generated packets were dropped due to buffer overflow for both device and router. On the other hand, as the traffic rate increases the number of packets dropped during the execution for both $\lambda = 5$ p/s and $\lambda = 10$ p/s also increases. Therefore, the queueing delay will also increases and thus the whole e2e delay.

VI. Discussions

We have presented an approach for obtaining a Markov chain modelling the network behavior, in particular, we have obtained a Markov chain model for the standard IEEE 802.15.4 protocol. We have shown that with our approach it is possible to find a realistic model which takes into account many aspects not covered by existing theoretical models. We applied our approach for a tree topology and we have estimated the e2e delay in two hops from the empirical Markov chain and the framework presented in [6]. Based on the results we can see that our approach is suitable for estimating the e2e delay in a multi-hop environment, a limitation of most of the existing theoretical models. It is necessary to mention here the limitations of our approach. One of the most important points in the methodology is the one related to the states identification. Therefore, it is necessary to deeply analyse the protocol specification in order to identify all the existing states and transitions and to avoid missing information. This is not a trivial task and means that we should have a comprehensive knowledge of the protocol behavior. Another limitation of our approach and contrarily to other theoretical models, is the fact that the obtained Markov chain models depend strictly on the input parameters. Previous Markov chain model are bound to specific parameters such as the chosen queue size, packet arrival rate, duty cycle, transmission power, etc. Changing the input parameters will give us another Markov chain model where, for instance, transition probabilities won't be the same as the ones found for some other configuration of parameters and some states that were not present previously may appear as a result of this new configuration. However, one of the major difficulties in modelling this kind of problem is to find the transition probabilities between the identified states. With our approach we are able to bypass this difficulty by obtaining a complete probability transitions matrix P for each state in the protocol. Moreover, from P we would be able to obtain the stationary distribution vector $\vec{\pi}$ of the system, an important result in order to estimate some other performance parameters.

VII. Conclusions & Future Work

In this paper we have presented a new approach for extracting empirical Markov chain models from network protocol traces by means of Process Mining techniques. An empirical Markov chain model was obtained for the standard IEEE 802.15.4 mac protocol allowing us to estimate the e2e delay for a multi-hop scenario. The contributions of our work can be enumerated as follows:

- We are able to obtain a Markov chain model from any protocol by analysing the protocol output (traces). Since this is an empirical approach we think that the obtained model is more realistic and accurate for

representing the exact behavior of the protocol with regard to the theoretical models (as the one proposed in [8]) encompassing all phenomenons introduced by the underlying operating system described in [3].

- The obtained Markov chain together with the sejourn time on each state allowed us to define the matrix A. By means of A we can compute the set of vectors $\vec{A}^r_{i,d}$ for $r = \{2, 3, 4....\}$, and then we were able to estimate the e2e delay from source to destination. We have mentioned that, in general, existing mathematical models are conceived for a star topology where delay and other performance parameters are found for the case of 1-hop transmission and the extension to include multi-hop transmissions is not trivial. With our approach we overcome this problem by proposing a way for estimating the e2e delay in multi-hop networks.

As a future work, we plan to extend the approach to some others protocols such as iQueue-MAC [10] and ContikiMAC [4], both of them are self-adaptive duty cycled MAC protocols. We also expect to set up a scenario by varying the distance between nodes in order to show how our approach takes into consideration the capture effect.

REFERENCES

[1] IEEE Standard for Information Technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - specific requirement Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). Technical report, 2007.

[2] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE J.Sel. A. Commun.*, 18(3):535–547, September 2006.

[3] Francois Despaux, Ye-Qiong Song, and Abdelkader Lahmadi. On the gap between mathematical modeling and measurement analysis for performance evaluation of the 802.15.4 mac protocol. In *RTN13*, 2013.

[4] Adam Dunkels. The ContikiMAC Radio Duty Cycling Protocol. Technical Report T2011:13, Swedish Institute of Computer Science, December 2011.

[5] Jan-Hinrich Hauer. Tkn15.4: An ieee 802.15.4 mac implementation for tinyos 2. TKN Technical Report Series TKN-08-003, Telecommunication Networks Group, Technical University Berlin, March 2009.

[6] 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.

[7] Joachim Herbst and Dimitris Karagiannis. Workflow mining with inwolve. *Comput. Ind.*, 53(3):245–264, April 2004.

[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*, page 18, 2009.

[10] Ye-Qiong Song. Réseaux de Capteurs Sans Fil : Comment Fournir La Qualité de Service Tout En Economisant l'Energie ? In INP Toulouse, editor, *Ecole d'été temps réel 2013*, Toulouse, France, August 2013. IRIT Toulouse.

[11] W.M.P. van der Aalst, A.J.M.M. Weijter, and L. Maruster. Workflow mining: Discovering process models from event logs. *IEEE TKDE*, 16:2004, 2003.

[12] Boudewijn F. van Dongen, Ana Karla A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and Wil M. P. van der Aalst. The prom framework: A new era in process mining tool support. In *ICATPN*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer, 2005.

[13] Gabriel M. Veiga and Diogo R. Ferreira. Understanding spaghetti models with sequence clustering for ProM. In *Business Process Management Workshops*, page 92103, 2010.

[14] Yipeng Wang, Zhibin Zhang, Danfeng Daphne Yao, Buyun Qu, and Li Guo. Inferring protocol state machine from network traces: A probabilistic approach. In *Proceedings of the 9th International Conference on Applied Cryptography and Network Security*, ACNS'11, pages 1–18, Berlin, Heidelberg, 2011. Springer-Verlag.

[15] Yunbo Wang, Mehmet C. Vuran, and Steve Goddard. Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. *IEEE/ACM Transactions on Networking*, 20(1):305–318, February 2012.