Energy Map Model for Software-Defined Wireless Sensor Networks

Gustavo A. Núñez and Cintia B. Margi

***Abstract*— Wireless sensor networks (WSN) is a technology commonly used for remote monitoring, tracking and detection applications. Since energy efficiency is a major concern in WSN, to know the remaining energy on each node could optimize the energy consumption. We propose a novel method to create an energy map for a Software-Defined Wireless Sensor Network (SDWSN). We implement an energy consumption prediction model into the controller, which obtains information of each node and estimates its energy consumption rate. This approach reduces the node processing overhead and memory usage when compared to other approach available.**

***Keywords*— Wireless Sensor Networks, Software-Defined Networking, energy map.**

I. Introduction

Wireless Sensor Networks (WSN) are formed by interconnected nodes that are able to sense characteristics from the real world, to process this information and to exchange it with other nodes into the network. These characteristics make WSN suitable for applications in different areas such as health, industrial and environmental monitoring, battlefield and residence surveillance. On the other hand, WSN have important constraints: low bandwidth, limited processing power and storage capacity, low communication range and limited energy [5].

Despite the current advances in Micro-Electro-Mechanical Systems (MEMS), energy efficiency is still a key problem in WSN. One of the main reasons is because sensor nodes depend on batteries to operated. Furthermore, some applications deployed the WSN in hostile environments where the maintenance becomes a challenging task. In order to address such issues, several approaches can be found on the literature. One approach is to reduce the energy consumption of the sensor nodes and of the network as a whole [1]. Another approach is to use the available energy as metric for the protocol by design, and thus improve the network lifetime. SPIN [12] and GEAR [24] use remaining energy as routing metric, while FLEECH [19] and RECHS [3] use energy in the decision algorithm to select the cluster head.

Besides these approaches, the sink could benefit from knowing the energy of network, and thus could predict future failures in the network due to depleted energy, or select the most adequate set of nodes to execute a given task. Thus, an energy map is defined as the information of the residual energy on each part of the network [18].

Gustavo A. Núñez and Cintia B. Margi¸ Department of Computer Engineering and Digital Systems, Universidade de São Paulo, São Paulo, Brazil, E-mails: gnunez@larc.usp.br, cintia@usp.br.

The energy map construction has two main steps that are performed by the sensor nodes. First the node estimates its energy consumption and remaining energy. Second the node informs its remaining energy to the node which is in charge to collect the data to construct the map. The constant execution of both steps increases the energy consumption, since there are more tasks related to obtaining the remaining energy and transmitting this information. Since transmission is the most energy costly task, one approach to reduce the network traffic caused by these updates is to use energy consumption prediction models [18]. One drawback from such approach is the increase in the processing overhead in the node.

Software Defined Networking (SDN) is a paradigm that separates the control plane from the data plane. In order to accomplish this separation, it has a programmable controller which is in charge of control logic decisions and the network devices become forwarding devices. This scheme allows the controller to have a global view of the network, giving the opportunity to adequate the routing algorithms [16]. McKeown et al. proposed OpenFlow [17] in 2008, the first southbound protocol to establish communication between an SDN controller and forwarding devices, focusing on wired networks.

Applying the SDN paradigm into WSN is known as Software Defined Wireless Sensor Networks (SDWSN), and it has been seen as a solution for inherent problems of the latter. Energy consumption, network management, mobility, interoperability and security are among the fields where SDWSN is being studied [15]. There are already several proposals in the literature [15], including TinySDN [20] and SDN-Wise [10]. The SDWSN controller normally runs in a station with higher processing power and without energy constraints. Some works require to merge the controller with the sink node [9], while others consider them different entities [20].

Given the centralized approach in the SDN paradigm, the network controller use information concerning available energy to improve route definition, either by using it as a metric or by using the information to avoid low energy nodes in a given route. Some SDWSN proposals mention the remaining energy in the nodes as an important information, but do not explain how to obtain it. Furthermore, the construction of an energy map in SDWSN has not been considered yet.

The main contribution from this work is a novel method to construct an energy map into a SDWSN. We follow the Markov chain approach proposed by Mini *et al.* [18], but instead of running the prediction model on each node, the controller obtain information of each node behavior and estimates an energy consumption rate to create and constantly update the

energy map. In that way, by centralizing the estimation we can reduce the processing overhead and the memory usage in the sensor nodes.

The remainder of this paper is organized as follows. Section II reviews the related work, while Section III explains the mathematics behind the energy consumption prediction model. Section IV describes the implementation characteristics. Section V shows the performance evaluation results that are discussed in Section VI. Finally, conclusions and future work are presented in section VII.

II. Related work

Different approaches have been explored to estimate the energy consumption in WSNs and predict the lifetime of the network. Mini *et al.* [18] compare three different approaches to predict the energy consumption and construct the energy map. The first one is a naive approach, where the node periodically sends its current energy level to a monitor node. The second one uses a Markov chain model to forecast the energy consumption; then the node sends its remaining energy and the energy consumption rate to the monitor node. The third approach use an Autoregressive Integrated Moving Average (ARIMA) model, in which the node calculates the parameters of the model and then sends its remaining energy and those parameters to the monitoring node. For the second and third approach the authors propose an error threshold. The node calculates the error between the last prediction sent to the monitor node and the new prediction. If this error is larger than the threshold, it sends a new message to update the parameters. The paper compares the accuracy of the estimation and the traffic overhead of the three approaches. The energy consumption model for this experiment has four operational states and the simulation was conducted using the software NS2. This paper has two main conclusions. First, the Markov chain and the ARIMA model approaches have a better energy efficiency than the naive approach. Second, the Markov chain model produces lower traffic overhead than the ARIMA model.

Han *et al.* propose IDSEP [11], which is an intrusion detection scheme based on energy prediction. Their energy consumption prediction model is based on Markov chains, similar to the Markov chain model in [18], but using five states instead of four. Another difference is that IDSEP runs the prediction model on the sink instead of on each sensor node. The model is implemented using NS2.

Peng Hu *et al.* [13] present a method to predict the energy level of the WSN based on the Hidden Markov Model (HMM). The proposal works with four energy level states and has a training process and a decoding process. The training process runs to calculate an approximation for the transition matrix and the emission probabilities. The decoding process runs to estimate the current state. The paper presents results from the proposal simulation using NS2.

Shi, Z. S. *et al.* [23] present an approach based on generalized stochastic Petri nets (GSPN). The main idea is to simplify the node behavior in two states: Active and sleep. The paper presents results from a network simulation and compares the energy consumption with the prediction obtained with the model. The simulation was conducted with TinyOS and Tossim.

Kerasiotis *et al.* [14] propose a method to predict the battery lifetime for a WSN platform based on a AA alkaline battery depletion profile. They show the results of their prediction method for different load profiles and use the TelosB platform for the implementation.

Gallucio *et al.* [10] and Akram *et al.* [2] propose SDN-enabled architectures for WSN. Those proposals include the battery remaining energy as a metric or important information for its operation. This information is include in the replies for the topology discovery protocol, but authors do not explain how such information is obtained.

While there are some SDWSN proposals that regard the information about remaining energy, the construction energy map is an issue that has not been considered. In this work we propose a method to create an energy map in a SDWSN, using the controller as and then we compare it with a similar approach used in WSN.

III. Energy consumption prediction model

Sensor nodes can be modeled as a device with different states of operation, such as: processing, transmitting, listening, sensing, and different low power modes. Therefore, it is possible to model each sensor node as a Markov chain, where each operation mode can be represented as a state of the Markov chain.

We define that the relation $X_n = i$ represents a node in a operation mode i at time-step n, and the probability that the next state be j can be represented as P_{ij}.

For a Markov chain defined by M states, the probability that a node in the state i will be in state j after n-step transitions is given by the *Chapman-Kolmogorov equation*, and is defined as:

$$P_{ij}^{n} = \sum_{k=1}^{M} P_{ik}^{r} P_{kj}^{(n-r)} \qquad for \;\; 0 < r < n \tag{1}$$

For a current state $i(X_0 = i)$, the total of time-steps a node will remain in state s during T time-steps is defined as:

$$\sum_{t=1}^{T} P_{is}^{t} \tag{2}$$

Then if a node is in state i and E_s is the energy dissipated during one time-step in state s, thus the expected amount of energy the node will spend in the next T time-step is expressed in the equation (3):

$$E^{T} = \sum_{s=1}^{M} (\sum_{t=1}^{T} P_{is}^{t}) E_s \tag{3}$$

In a system with M states, the transition probabilities among all states are represented by a MxM matrix. Thus, to forecast the energy consumption following the equation 3, the probability P is substituted by the probability matrix. The 4 states used in our model are: transmitting, listening, processing, and low power mode.

IV. Implementation and Simulation

The energy map was implemented and tested using IT-SDN [4]. IT-SDN is an SDWSN framework based on TinySDN [20]. Its architecture has three components: southbound protocol, neighbor discovery protocol, and controller discovery protocol. The southbound protocol defines the communication between the controller and SDN-enabled devices. The neighbor discovery protocol is in charge of obtaining and maintaining nodes neighborhood information. The controller discovery protocol determines a next hop candidate to reach the controller. Current IT-SDN implementation runs on Contiki 3.0 [7].

Next we describe the prediction model, the two implemented schemes, and the simulation methodology.

A. Prediction model

The prediction model algorithm implemented is depicted in Figure 1. As the probability matrix depends on the network behavior, which may change over time, the first step is to construct a transitions matrix. The transitions matrix has the information about how many times the node goes from one state to another, and how many time-steps it remains steady on each state. Then, using the transitions matrix and the total transitions counted for a certain period, the node calculates the probability matrix dividing each row by the total transitions of each state. The time the node remains constructing the transitions matrix is defined as a model parameter. Lastly, the node uses equation 3 to forecast the energy consumption and to calculate the consumption rate for the time-steps required. The consumption rate calculated is compared with the last consumption rate registered, and if the difference is higher than a limit defined, the program registers the new consumption rate.

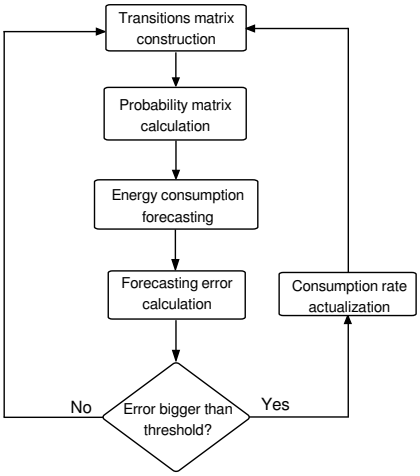


Fig. 1: Energy consumption prediction model algorithm

In our implementation, we construct the transitions matrix using Energest [8]. Energest is a Contiki tool designed to count the time each state remains on. When the node goes from one state to another, Energest stops the first state timer and starts the second state timer. We use this information to count the transitions among all states and the total transitions of each state.

B. The two implemented schemes

We implement the energy map using two different schemes: Scheme 1, which is based on previous works; and Scheme 2 that is our proposal. In Scheme 1, each node processes its own prediction algorithm and sends its consumption rate and remaining energy to the controller. The controller constructs the energy map and uses the consumption rate to update it periodically. In Scheme 2, each node constructs the transitions matrix, calculates the total transitions for each state, and sends them and its remaining energy to the controller. With this information the controller executes the forecasting algorithm and calculates the consumption rate of each node. Then, it constructs the energy map and uses the consumption rate to do periodical updates. For both cases, each node is monitoring its behavior. When detecting an important change on it, the node sends an update to the controller, if necessary.

In this manner, Scheme 2 reduces the processing on the node but sends messages with larger payload than Scheme 1. Since our prediction model uses 4 states and the message includes the transitions matrix and the total transitions of each state, the message payload has 80 bytes. On the other hand, Scheme 1 message payload has 8 bytes.

C. Simulation methodology

The topology used for all experiments is formed by one controller, one sink, and four sensor nodes; deployed as shown in Figure 2. Nodes number 3, 4, 5, and 6 are sensor nodes that send one message to the sink every 60 seconds.

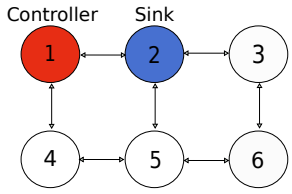


Fig. 2: Simulations topology

The transitions matrix updates was tested for two different periods: 10 minutes and 20 minutes. Other parameters are shown in Table I.

TABLE I: Forecasting simulation parameters

Parameter	**Value**
Forecasting	
Time-step	10 ms
Forecasting	100 time-step
Error threshold	5 %
Energy Consumption	
Processing (4 MHz)	2,33 mA
Low Power Mode	0,180 mA
Transmitting (0dB)	21,7 mA
Listening	22,8 mA

The simulation experiments were conducted using COOJA [21] from Contiki 3.0. The platform emulated was TelosB and the energy consumption calculations are based on the mote datasheet [6] and the results shown by Prayati et.al [22]. The main reason to follow Prayati's work is because it presents

several information about the TelosB hardware consumption for different configurations, such as: CPU clock frequency, low power mode (LPM) operation, and radio module consumption for different transmission gains.

The Schemes comparison was conducted by analyzing the energy consumption and the memory space on each sensor node. The energy consumption is analyzed by the processing time and the radio usage.

V. Results

Table II shows the memory usage in the sensor node for three different cases. The first case shows the memory usage by the IT-SDN enabled-node module and a simple application which generates the data traffic. The second case shows the memory usage when adding the prediction model for the Scheme 1. The last case is when using the prediction model for the Scheme 2. The results show that using Scheme 2 the node saves 1,2 kB of memory, which represents 41% of the prediction model code for Scheme 1.

TABLE II: Code size on the sensor node including IT-SDN and both energy consumption prediction model schemes

Scheme	ROM	RAM	Total
Memory usage without prediction model	37944	8222	46358
Memory usage for Scheme 1	40556	8474	49246
Memory usage for Scheme 2	39336	8484	48036

Figure 3 and 4 show the results when comparing the energy consumption for both schemes and update periods. Figure 3a and Figure 3b present the results of processing energy consumption when updating the transition matrix every 10 minutes and every 20 minutes. In the same manner, Figure 4a and Figure 4b present the results of the radio module energy consumption for both periods.

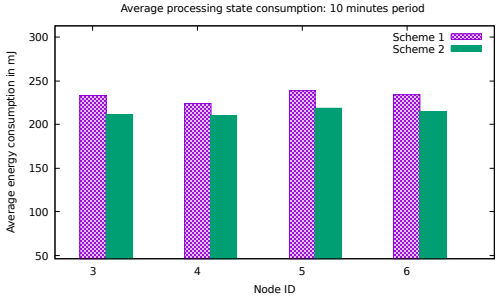
Table III presents the energy balance when comparing both schemes and for both update periods. The information on *Processing* and *Radio* columns shows the difference of energy consumption of Scheme 1 and Scheme 2. When the number is positive, it means that Scheme 2 had a better performance. When the number is negative, it means that Scheme 1 had a better performance. The *Total* column shows the final balance of each node, following the same signs rule.

TABLE III: Energy consumption balance

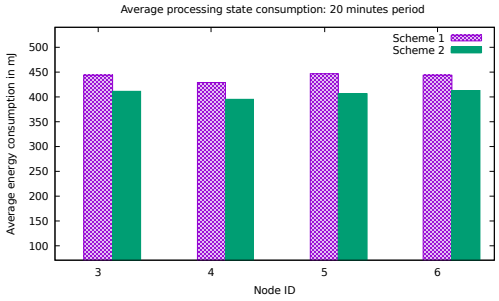
Node ID	10 minutes update		Total (mJ)	20 minutes update		Total (mJ)
	Processing (mJ)	Radio (mJ)		Processing (mJ)	Radio (mJ)	
3	22,71	-10,89	11,82	32,74	-28,02	4,72
4	13,70	-12,19	1,51	32,56	-24,57	7,99
5	21	-19,93	1,07	41,26	-31,07	10,19
6	18,5	-1,04	17,46	34,09	-28,49	5,6

VI. Discussion

The results presented in Section V show that Scheme 2 code is 41% smaller than Scheme 1 code. By analyzing the memory space in the TelosB mote (it has 48 KB of ROM and 10 KB of RAM), the IT-SDN code takes up the 79% of the total ROM and the 82,2% of the total RAM. Thus, the remaining 21% and 17,8% are available for applications, including the prediction code. From this point of view, the prediction code reduction means a 12% more ROM for other applications.



(a) Average energy consumption using 10 minutes prediction periods



(b) Average energy consumption using 20 minutes prediction periods

Fig. 3: Processing state energy consumption

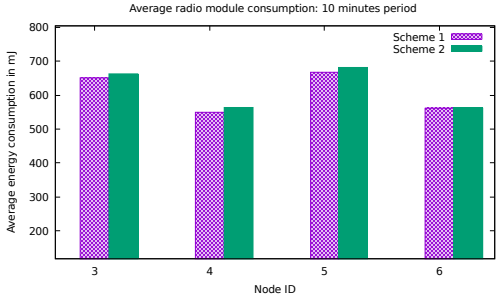
The processing time measurements also show a reduction when using the Scheme 2. The reduction was around 8% in all the sensor nodes and for both transition matrix update periods. This fact, and the low coefficient of variation, give us an idea that the measurements were poorly affected by the other processes running on the node. It also shows a low dependency between the processing time reduction and the node position in the topology.

In the radio usage case, it increased its energy consumption in all nodes when using the Scheme 2. This result was expected due to the great difference in the message payload size. Nodes 3 and 5 were the ones with higher increase on the energy consumption, since they have to forward messages from node 6 to the controller.

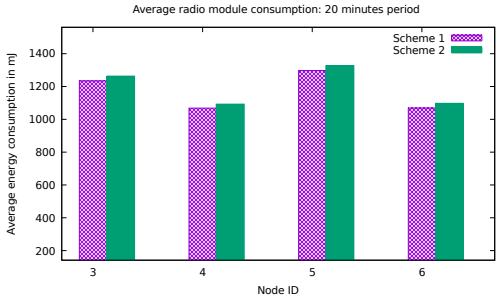
Finally, the energy consumption balance reveals a small improvement in the sensor node performance when using Scheme 2. This means that the reduction in the processing when implementing the Scheme 2 compensates the radio usage increased. Thus, we were able to decrease the processing in the node and the memory space, as well as the energy consumption.

VII. Conclusions and Future Work

In this work we introduce a model to create an energy map on a SDWSN. This model uses an energy consumption



(a) Average energy consumption using 10 minutes prediction periods



(b) Average energy consumption using 20 minutes prediction periods

Fig. 4: Radio module energy consumption

prediction model running in the network controller instead of running it on each node, as previous works proposed.

The results show that our proposal reduces the processing time and requires less memory space in the node when compared to other approach on the literature. A drawback of the model is the increase of the radio module usage. Even though, the model achieved a reduction in the overall sensor node energy consumption.

For future work, we intend to expand the energy consumption model by adding more states and including a battery model. Our goal is to implement a realistic energy consumption model that to create a reliable energy map could be used by the controller for different tasks.

ACKNOWLEDGMENTS

C. B. Margi is supported by CNPq research fellowship #307304/2015-9. G.A.Núñez is supported by a scholarship from CAPES-PROEX #0212083, and by the University of Costa Rica.

REFERENCES

[1] Anastasi, G., Conti, M., Di Francesco, M., Passarella, A. (2009). Energy conservation in wireless sensor networks: A survey. Ad hoc networks, 7(3), 537-568.

[2] Akram, H., Gokhale, A. (2016, May). Rethinking the Design of LR-WPAN IoT Systems with Software-Defined Networking. In Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on (pp. 238-243). IEEE.

[3] Alassery, F., Ahmed, W. K. (2016, September). Smart wireless sensor networks powered by remaining energy cluster head selection protocol. In Sarnoff Symposium, 2016 IEEE 37th (pp. 59-64). IEEE.

[4] Alves, R., Oliveira, D., Núñez, G., Margi, C. (2017). IT-SDN: Improved architecture for SDWSN. XXXV Brazilian Symposium on Computer Networks and Distributed Systems, 2017.

[5] Culler, D., Estrin, D., Srivastava, M. 2004. Guest Editors' Introduction: Overview of Sensor Networks. Computer 37, 8 (August 2004), 41-49.

[6] Datasheet, T. Available Online: `http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb_datasheet.pdf` (accessed on 5 april 2017).

[7] Dunkels, A., Grönvall, B., Voigt, T. (2004, November). Contiki-a lightweight and flexible operating system for tiny networked sensors. In Local Computer Networks, 2004. 29th Annual IEEE International Conference on (pp. 455-462). IEEE.

[8] Dunkels, A., Osterlind, F., Tsiftes, N., He, Z. (2007, June). Software-based on-line energy estimation for sensor nodes. In Proceedings of the 4th workshop on Embedded networked sensors (pp. 28-32). ACM.

[9] De Gante, A., Aslan, M., Matrawy, A. (2014, June). Smart wireless sensor network management based on software-defined networking. In Communications (QBSC), 2014 27th Biennial Symposium on (pp. 71-75). IEEE.

[10] Galluccio, L., Milardo, S., Morabito, G., Palazzo, S. (2015, April). SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks. In Computer Communications (INFOCOM), 2015 IEEE Conference on (pp. 513-521). IEEE.

[11] Han, G., Jiang, J., Shen, W., Shu, L., Rodrigues, J. (2013). IDSEP: a novel intrusion detection scheme based on energy prediction in cluster-based wireless sensor networks. IET Information Security, 7(2), 97-105.

[12] Heinzelman, W. R., Kulik, J., Balakrishnan, H. (1999, August). Adaptive protocols for information dissemination in wireless sensor networks. In Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking (pp. 174-185). ACM.

[13] Hu, P., Zhou, Z., Liu, Q., Li, F. (2007, May). The HMM-based modeling for the energy level prediction in wireless sensor networks. In 2007 2nd IEEE Conference on Industrial Electronics and Applications (pp. 2253-2258). IEEE.

[14] Kerasiotis, F., Prayati, A., Antonopoulos, C., Koulamas, C., Papadopoulos, G. (2010, July). Battery lifetime prediction model for a wsn platform. In Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on (pp. 525-530). IEEE.

[15] Kobo, H. I., Mahfouz, A. M., Hancke, G. P. (2017). A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements. IEEE Access.

[16] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., Uhlig, S. (2015). Software-defined networking: A comprehensive survey. Proceedings of the IEEE, 103(1), 14-76.

[17] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 38(2), 69-74.

[18] Mini, R. A., Loureiro, A. A., Nath, B. (2003, September). Prediction-based energy map for wireless sensor networks. In IFIP International Conference on Personal Wireless Communications (pp. 12-26). Springer Berlin Heidelberg.

[19] Nayak, B. K., Mishra, M., Rai, S. C., Pradhan, S. K. (2014, December). A novel cluster head selection method for energy efficient wireless sensor network. In Information Technology (ICIT), 2014 International Conference on (pp. 53-57). IEEE.

[20] de Oliveira, B. T., Margi, C. B., and Gabriel, L. B. (2014). TinySDN: Enabling multiple controllers for software-defined wireless sensor networks. In IEEE LATINCOM.

[21] Osterlind, A. Dunkels, J. Eriksson, N. Finne., T. Voigt. Cross-Level Sensor Network Simulation with COOJA. Proceedings. 2006 31st IEEE Conference on Local Computer Networks, Tampa, FL, 2006, pp. 641-648. doi: 10.1109/LCN.2006.322172.

[22] Prayati, A., Antonopoulos, C., Stoyanova, T., Koulamas, C., Papadopoulos, G. (2010). A modeling approach on the TelosB WSN platform power consumption. Journal of Systems and Software, 83(8), 1355-1363.

[23] Shi, Z. S., Wang, C. F., Zheng, P., Wang, H. Y. (2010, December). An energy consumption prediction model based on GSPN for wireless sensor networks. In Computational and Information Sciences (ICCIS), 2010 International Conference on (pp. 1001-1004). IEEE.

[24] Yu, Y., Govindan, R., Estrin, D. (2001). Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks.