Semantic network coding for WSAN in IoT Applications

Sylvain Cherrier, Rami Langar

Semantic network coding for WSAN in IoT Applications

Sylvain Cherrier*, Rami Langar†
LIGM, Univ Gustave Eiffel, CNRS, ESIEE Paris, F-77454 Marne-la-Vallée
Email: *sylvain.cherrier@univ-eiffel.fr, †rami.langar@univ-eiffel.fr

***Abstract*—Software uses and approaches in the Internet of Things (IoT) are very varied and rich. But the data they collect is often gathered from networks of sensors. However, the constraints of Wireless Sensors and Actuators Networks (WSAN) are not always well taken into account by IoT software approaches. Thus, the massive data collection, well managed by Big Data tools in IoT applications, conflicts with the energy constraints to which the sensors are subject. Each transmission is costly for these devices and a massive stream reduces the whole WSAN lifetime.**

This article presents an approach that meets the data needs in the vision of IoT users while seeking to limit the impact of the massive transmissions required. Edge computing is often used to reduce the amount of data to be treated in this research field. But the reduction is still made outside the WSAN, leaving a significant transmission load for these constrained networks. Energy-aware IoT applications get increased attention from the researchers. Network coding is an appropriate and already well-known tool for saving energy in constrained networks. Then the question is how to combine all the messages to reduce their number while maintaining their accuracy.

In this paper, we present an approach focusing on the gain obtained thanks to the semantics of data. The Semantic Network Coding leads to a reduction in the number of messages forwarded, thus leveraging the network and saving energy, while keeping a good quality to the collected data.

***Index Terms*—IoT, Semantic, Network coding, WSAN**

I. Introduction

The Internet of Things (IoT) is a major trend in network research. It combines the large installed base of public protocols for network interconnection (*Internet*) with recent developments in the field of sensor networks, which have become a major part of the objects of loT (*Things*). With strong demand from the industry, the field is very active and the angles of analysis very diversified.

The usual solutions in the proposed approaches are usually based on Cloud/Big Data. Very large amount of data are collected by objects (sensors in WSAN), and sent to the Cloud where it is stored. Then applications use BigData tools to create analysis and extract representations whose objective is to help users manage their buildings, business or cities.

In this vision, the data stream is always coming from the Things, which are mainly Wireless Sensors and Actuators, and send to the Cloud. This centralized architecture has some issues, in terms of ownership, privacy, and amount of data. Another way to handle the data stream is to distribute the treatment. As the content of each message in Things is rather small and specific, specialized in a limited or unique value, it appears that some pre-treatment can be done before transmitting the data to the central point.

The idea of distributed computing for IoT has been presented in many papers, under different names such as Edge computing, Cloudlet or Fog computing, depending on the place and the actor in charge of the treatment. In Edge computing, a trade-off is made between the computing capability of a device and its proximity to the place where the data was collected. The solution is to limit the number of hops, and to control the amount of basic information stored in the cloud. Data are treated as closely to the sensors as possible, and as soon as possible. But it is very rare that the treatment has been considered to be done directly on the nodes, because the limited processing power of IoT devices are not able to handle big processes.

In fact, WSAN devices have poor processing capabilities and small memory, because they were intend to mainly create and receive network messages. But as the main constraint of these devices is energy consumption, it appears that data transmission is one of the most costly operations they offer. And even with its limited processing capabilities, we assume that the device has enough to manipulate, adapt the data and then limit the need for transmission. We also assume that the treatments are not that complex, because the nature of the data manipulated is simple (in size, value and type) and the computation easy.

In this paper, we present our proposition to use the semantics of the collected data, assuming that it is fairly well defined and not very variable within a given WSAN. In Section II, we present the energy issue in WSAN/IoT, the contribution of network coding and the interest for semantic in IoT. Section III describes our approach with its pros and cons, while Section IV gives details on our experimentation. Finally, we conclude and give some propositions for the evolution of this work.

II. Related Work

In the IoT field, precisely in Massive IoT, there are two important issues to solve. On one side, the huge amount of data to be treated by Big-Data solutions, and on the other side, the difficulty to send these data from the very constrained devices used in IoT. With a large number of devices, the pricing leads the manufacturers to design them with low processing power, memory size, low network range and throughput to reduce the total cost of the installed hardware. These devices

must be very cheap (because of the number to be used), the energy consumption very low (because the maintenance must be reduced to the minimum). On the other hand, users want them to send a lot of information, which goes against the previous constraints.

One idea that we have already explored in previous papers [7] [8] is to limit energy consumption by computing data right in place, within the device. As IoT devices offer some computing capabilities, our proposition is to try to maximize the use of the processing power instead of transmission, as the energy cost of computing is often less than sending/receiving data (see Tables I and II). In this approach, the discrete values sensed by the device are lost. Depending on the use-case, it can make sense. Some IoT applications propose to react to events and not to store data for a asynchronous post-analysis (event-centric approach instead of data-centric). Y. Sun et al. use a similar idea with a rule-engine on the devices filtering a "majority of the unused atomic event" [16].

F. Xhafa et al. have explained in [18] that "*processing and aggregating large data generated at IoT layer before sending*" give good results for IoT applications. The authors argue that this solution "*achiev[es] low latency*", and "*enhanc[es] privacy, security*". We also invoked the same argument in a previous paper [6].

H. Nasiri et al. show in [11] that ("*Internet of Things (IoT) technologies enable cities to obtain valuable intelligence from a large amount of real-time produced data*"). Smart city is one of the domain of the IoT which is main concerned with this solution. In fact, there is a main difference between the network flow of the Internet of Data[1]. The data stream in the Internet of Data is more versatile, irregular and diversified. In the Internet of Things (and more precisely, from the devices of a WSAN that feeds the IoT), the kind of data is less different, less complex, and more repetitive. These are specificities that can be useful from this "*Continuous streams of unstructured data*" [11] with a "*lack of semantic interoperable standard*" [5].

T. Li et al. also present Network coding in the IoT in [10] "*Network Coding (NC) can increase the robustness and throughput of wireless networks*", but their purpose is mainly based on encryption and security.

G. Peralta et al. argue [14] in the same direction on security. They also show the reduction in terms of delay in the network, and the improvement on the energy consumption. On the other hand, the authors warn against "*The non-negligible increase in packet size and computational cost*" of this kind of solutions. However, it seems that the way IoT applications work focuses mainly on the network capabilities of IoT devices, and not the contribution of their processors which remains underused.

There is a need for a better management of the network load. F. Safara et al. argue [15] that "*Most IoT applications focus on monitoring discrete events that generate an excessive amount of data*". In their survey about network coding and reliability in WSAN [3], E. Al-Hawri et al. explain that "*Energy efficiency is the main concern rather than bandwidth, since nodes typically produce small volumes of data*". Increasing reliability means to work primary on energy. To reach that goal, the authors present different solutions based on the nodes placement, and on network coding.

C.HS. Oliveira et al. from our lab have also presented a proposal for network coding in wireless network [12], but the implementation of their solution is too much a burden for the limited processing capabilities of our devices (as shown above by G. Peralta et al. in [14]).

There are multiple solutions to select inside the network the set of elements that will be mixed. One approach is to define clusters. S. Balakrishna et al. describe different types of clusters [5] : Centroid base, connectivity based et density based, depending on the constraints the solution tries to solve. The idea of semantic network coding was presented by F. Xhafa et al. in [18]. In their paper, they describe a experiment in which the sensors generate raw data. Semantics (meta-information such as the "*geo-location information, timestamp, actor (device)*") will be added to these data. The network coding is made at the Edge level. In this paper, we propose to code at the node level, to avoid energy consumption on upper nodes in the tree.

S. Balakrishna et al. have also classified the information inside the network in different classes: "*Raw Data, Structured data, Perception data and Executive Data*" [5]. *Raw data* are gathered by the sensors (sound, temp, humidity). They are then structured (date, format, unit, source) in *Structured data*. The next Level (*Perception data*) adds the semantic (Context, concept). The highest level (*Executive Data*) is the application reasoning, with analysis, predictive and actions.

The idea of a Semantic IoT was described by A. Palavalli et al. in [13]. For the authors, the conception of an Ontology is mandatory to describe and to request in the large amount of data, once associated to concepts. They focus on the improvements from the application point-of-view, and the definition of their model. They didn't study the impact of the place where the metadata are added, neither the gain in energy consumption from the devices side, as their solution is mainly centralized, which is the common practice..

Our idea in this paper is to create semantic clusters, mapped on the network topology. In a given WSAN, the nodes sense the same kind of physical values, and all the nodes are known. The content format is known in advance, and the network organization too. WSAN devices are able to measure some data such as temperature, luminance, movement. Often, the list is short. We assume that there are not so much different kinds of data, so the semantics are limited. Within the WSAN, the device are quite similar (even if at the IoT application level, multiple and various WSAN and other Things are included).

In our proposal, the data will be semantically network coded directly inside the WSAN. This approach is not adapted to all IoT applications. Depending on the use-case, merging the values collected in the global result can be accepted, as in fact the goal of IoT applications can be real time reactions to events that are detected. However, in a smart city

[1]As the opposite of *Internet of Things*.

application monitoring traffic for example, the precise value of the number of cars on each specific road at a specific time may be required. But IoT applications are very varied, and this precision is not always useful. Y. Sun et al. argue that "*In real application scenario, [...] many messages may include the same event signal*" [16]. The cost of storing such detailed information must be justified by the treatment that the application will make of it. Y. Sun et al. assume that "*In a smart building system, sensors report data periodically*" [16] and their solution in which "*transmission messages [...] are the aggregated results*" "*reduce[s] the throughput of data transmission, thus saving energy effectively*".

III. Discussion

TABLE I
CC2420 Power consumption

Mode	Parameter	Comsumption
Receive	-25 dBm	8.5 mA
Transmit	-15 dBm	9.9 mA
Transmit	-10 dBm	11 mA
Transmit	-5 dBm	14 mA
Transmit	0 dBm	17.4 mA

TABLE II
CC2538 Power consumption

Mode	Parameter	Comsumption
CPU Running, no Radio	16 Mhz	7 mA
CPU Running, no Radio	32 Mhz	13 mA
RX mode	-50 dBm	20 mA
RX mode	-100 dBm	24 mA
TX mode	0 dBm	24 mA
TX mode	7 dBm	34 mA

In [6], we already have studied the impact on energy according to the design of the software architecture. As the WSAN devices are very constrained in energy, network usage must be taken into account. Our lab uses two kinds of devices based on Texas Instruments CC2420 [1] or cc2538 [2] chips. Tables I and II show the energy consumption on these two chips for different functions, from computation compared to the reception/transmission of the data. It shows that the cost of transmission is greater than processing. However, the computing capabilities of these nodes are limited, the clock is slow, and the memory small. But it is still possible to make some operations.

In our lab, we mainly use Contiki [9] to create IoT applications. The interest for Contiki is that it offers 6LowPan, RPL [17] and CoAP. It is then possible to mimic a REST architecture to exchange data between the WSAN and the outside world, forming the IoT application, while hiding from outside the characteristics of a IEEE 802.15.4 network.

In order to give the IPv6 access to each node, Contiki organizes the mesh WSAN network using the RPL routing protocol. With RPL (see Fig. 3), each node has its own IPv6 address, and is accessible from outside the network. RPL builds a tree inside the WSAN to give the illusion of a direct access to each node, while in fact there are multi-hops from the sink to the given node.

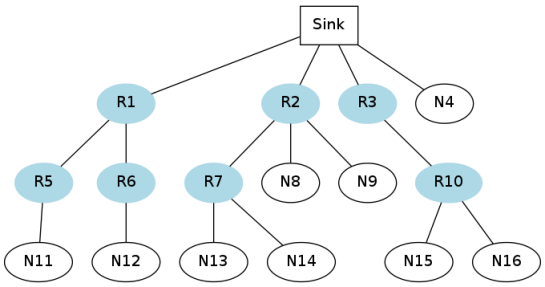

Fig. 1. In a RPL tree, the *sink* is the root of the tree and the node routing data outside the network. Nodes are accessible from the Internet with their IPv6 address (i.e. From outside, all nodes seem to be at the same level, but inside the WSAN, a message from *R7* will be forward by *R2*. It is also not visible from outside that the node *R7* is in charge of relaying messages to/from nodes *N13* and *N14*)

From the IoT application point-of-view, the WSAN is flat and all nodes are directly connected and accessible in IPv6. In fact, the WSAN is a multi-hop mesh network. Some nodes act as routers, forwarding data to others 1. Transmitting data has a cost for the node itself, but also for each node that is on the path to the sink. As the energy is one of the main concern for the researchers in WSAN, S.A. Alvi et al. have proposed in [4] some improvements in RPL for a better stability of the tree and a reduction of energy consumption. In our paper, we concentrate on the way the RPL tree can be used, and the impact of the data transmission by the nodes. In the tree T (such as the RPL tree presented Fig. 1) where each node $n \in T$:

- the $root$ of the tree is n_0.
- The $level$ of a node n is the distance from the sink n_0 (The sink n_0 is at level 0)
- the $height$ $\eta(T)$ is the longest path found in T
- the $size$ $|T|$ is the number of nodes of the tree
- the $pathlength$ $\pi(T)$ is the sum of the levels of each of the nodes in T
- the $subtree$ T_k is the tree having n_k as root.

When a node n_x sends data, the message is forwarded by all the nodes on the path to the sink. Each node n_x sends data, and forwards all the messages from each node of its sub-tree T_x. When all the nodes send data in a tree T, a total of $\pi(T)$ messages are transmitted within the network.

In our network presented Fig. 1, when the 16 nodes send one message each, there are 34 transmissions. The node $N2$ sends its own data, and is in charge of forwarding all the data from its sub-tree (5 nodes).

Each transmission/reception consumes energy (Table I and II), and the 4 nodes at level 1 represent 47% of the global consumption (Table III) (but only 25% of the total number of nodes). Looking at Fig. 1, we can see that node $R2$ has 6 nodes in charge. This node has a risk of consuming all its energy quickly, because of all the forward to do. As RPL is

TABLE III
MESSAGES BY LEVEL IN TREE FIG. 1

Level	Nb of msg	Nb of forward	Total	%
1	4	12	16	47%
2	6	6	12	35%
3	6	0	6	17%

dynamic, the tree can change after a while (as an important router node runs out of energy), i.e. $R7$ attaches to R1 and $N8$ and N9 to $R3$. But in that case, $R1$ will have 7 nodes to deal with, and will consume 7 times more energy than, for example, $N4$. The issue here is that the most important nodes (from the routing point-of-view) will be the first to run out of energy, stopping or endangering the network operation.

One idea to protect the network from this issue is to reduce the number of transmissions, with semantic network coding. Instead of transmitting its own data, and forward all the data coming from the sub-tree, the node mixes its own data with the upcoming message, reducing the number of transmission to a single one. At the end-point, the receiver has a unique message with a global value.

In our solution, nodes limit the number of messages they are forwarding. We assume that in this WSAN, the devices senses a unique, or a limited set of physical values (temperature, humidity, presence, luminance, etc). The semantics of the content are similar, or chosen in a small list. And in a multi-hop mesh network, we can consider than nodes in the same sub-tree are not far from each other.

As said above, the network tree is never very far from the real topology: I.e. in a smart city, the devices counting the number of cars on the road transmit the value to a router in the bock of buildings, that will forward the data to another router in the same neighborhood, and so on. So it makes sense to add the number of cars counted on each router to get the global traffic information in these city's district. Adding all the results of the different routers gives a global view of the traffic of this part of the city. The similarity in the collected data, and the proximity of the network organization compared to the topological position of the nodes argue for the use of a data computation to obtain a reduction of the number of transmissions.

This can be organized in multiple ways. With a unique kind of sensing (let's say temperature), each node can average its value with the values received from its children (see Fig. 2), and send this to its father (the average value, and the number of measures, as the weight of this average). At each level, the average is computed, and a unique message is transmitted to the upper level. The semantic network coding function here is average, but our solution proposes also to sum up or count values. This can be useful with traffic surveillance, or to count people in a smart building, a museum, etc.

If there are multiple kinds of measures, a vector of different values is transmitted, each node filling its part of interest. Here also, at each level of the tree, a computation is made, using average or sum of the different vectors received to create a new one to send to the upper level. Data computing costs less energy than sending it. The CC2420 or CC2538 devices offers enough computation power (CPU speed, memory space) to get the result.

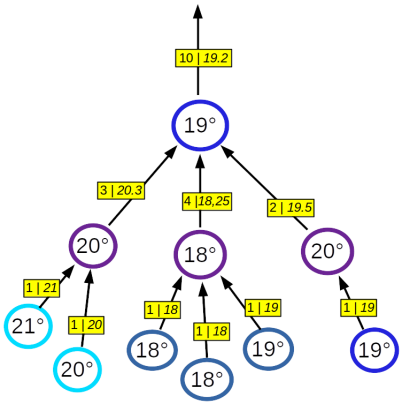

Fig. 2. In this example, the semantic network coding uses a simple vector containing the cardinality and the average temperature value. The leaves at the bottom of the tree send their sensed values. The first router mixes these value with its own and send the result to its router, and so on. The sensed and the calculated values may be stored on the node if necessary.

In our example Fig. 1, the semantic network coding reduces the number of messages from 34 to 16 (from $\pi(T)$ to $|T|$). If we assume that messages are short (only a value gathered by the sensors, or a vector with some values) and that the semantic network coded data still fits in one message, the global transmission reduction ratio R of the tree T can be calculated by the following formula: $R_t = \frac{|T|}{\pi(T)}$.

IV. EXPERIMENTATION

To test the feasibility of our proposal, we programmed an experiment for a 6LowPAN network. For this purpose, we chose Contiki-3.0 [9]. This free operating system runs on several devices that use the IEEE 802.15.4 protocol for wireless network. It implements 6LowPAN[2] to have IPv6 on these devices. With the routing protocol RPL, the IEEE 802.15.4 network is then organized in a cluster tree, making each node accessible with its IPv6 address without knowing the tree organization.

The Figure 3 shows the tree being build during experiment startup. Each node discovers its neighborhood and participates to the elaboration of the DODAG (Destination-Oriented Directed Acyclic Graph). The construction of this tree has a convergence time of few minutes. There are few restrictions to note. The tree may not match the topology of the environment, as each node choose its preferred parent according to a metric called LQL (Link Quality Level). Depending on the interference, a node can choose a different parent than the obvious one. Another characteristic of RPL is that the DoDAG is dynamic, and can change over times. But as mentioned above, we can assume that the network topology will be close to the geographical one.

[2]Acronym of IPv6 over Low -Power Wireless Personal Area Networks.

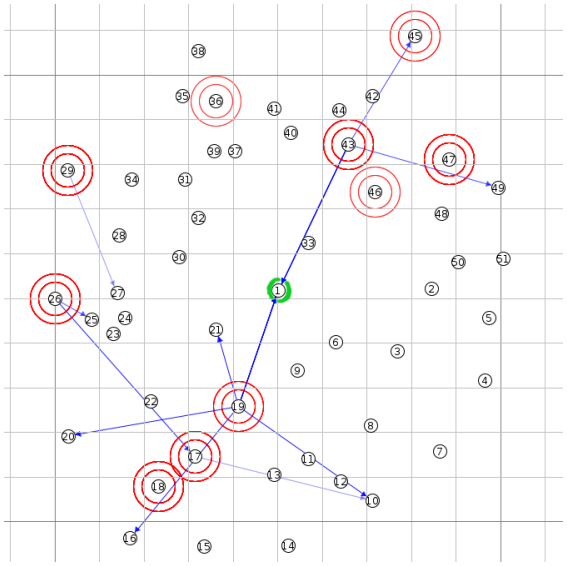

Fig. 3. State of the RPL DODAG in our Cooja simulation at 02:05. *Node 1* (in green) is the *sink* in the centre of the figure. The tree is visible, and *nodes 19, 17, 43* and *26* are in charge of forwarding messages to the *sink*. Each square represents 100m^2.

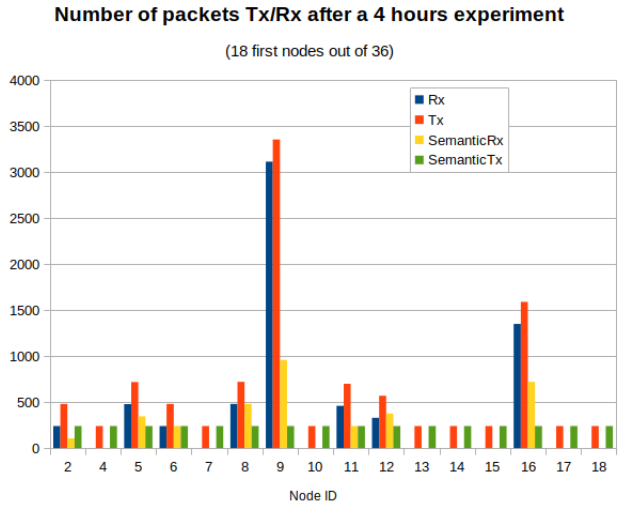

Fig. 4. this figure represents the number of packets received and transmitted for each node (the 18 first only) after a 4 hours experiment. The number of transmitted packets is the some for each node in the case of Semantic Network coding. On the contrary, in the standard organization, the router nodes are far more stressed.

TABLE IV
EXPERIMENTAL SETUP (SEE FIG. 3)

Parameter	Value
Radio range of the node	25 m
Network dimensions	130 × 130 m
node type	Zolertia Z1
radio chip	TI CC2420
Sink (RPL root)	1
Normal node	50

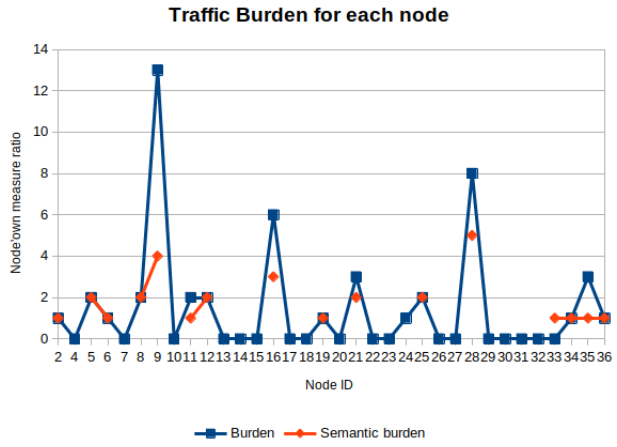

Fig. 5. With the Semantic Network Coding, each router reduces the number of forwarded packets. The traffic generated by its children decreases. In this Figure, Node 9 has less traffic to handle in the Semantic Network Coding version because its own children have reduced the number of packets to forward. The higher the router is in the tree, the more visible the reduction.

For this experiment, we used Zolertia Z1 devices. They are equipped with a CC2420 radio chip(see Table I). The micro processor is a MSP430f2617, consuming less that 10mA at 16MHz, and 0.5mA when running at 1MHz. The CC2420 consumes between 17 and 18mA in Rx/Tx mode. It has 92kB of ROM (to store the operating system, 6LowPAN stack, RPL protocol plus the soft we created) and 10kB of RAM to manipulate data (see Table IV).

For our simulation, we used Cooja, the network simulation/device emulator provided with Contiki-OS. It gives the programmer the ability to test the code in real conditions (as the device emulator strictly respects the constraints of each device) in a simulated network. The only difference is the software execution time in the devices emulated (depends on the server processor power, the number of node to emulate, and the soft complexity). For our experiment, this speed varies from 10% to 1000% of the real software speed. The other difference is the quality of the simulated radio network, which is flawless (which is not always realistic). Nevertheless, Cooja is a powerful tool for performing experiments.

In this experiment, we have one sink and 36 nodes (See Fig. 3 and Table IV) that sense physical data, and send them to the sink. Two versions of the experiment are proposed. In the **standard** version, each node sends a value to the sink. In the **Semantic Network Coding** version, each node sends the value to its father. The father calculates a new value (a mix of of all the values sent by its children) and transmits it to its own father.

As each router computes data and reduces traffic, activity at the higher levels of the tree decreases. Fig 5 shows the impact on each node in terms of data volume received. For example, the Node 9, very close to the Sink, receives from the sub-tree 13 times its own data volume. In the Semantic Network coding version, the same node sees its traffic load received reduced to 4 times its own.

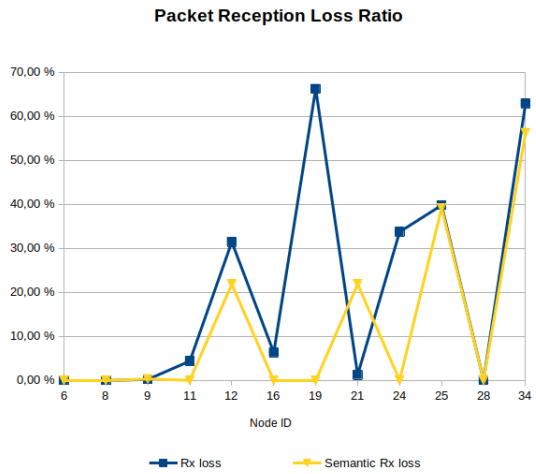

Fig. 6. Since the Semantic Network Coding reduces traffic, there is less packet loss. The router nodes still achieves a good transmission quality (Node 9 for example has a better loss ratio in normal organization, but with 4 times more packets to transmit). Node 19 has a important loss rate in the normal version, but is at the edge of the network, with only one child. However, the Semantic Network Coding has also some reception issues (e.g. Node 21).

Fig. 4 represents the traffic load of each node after a 4 hours experiment. With the Semantic Network Coding version, the outgoing traffic is equal for each node, saving energy for the router nodes (reception and transmission). In the standard version, nodes 9 or 16 are more heavily used for routing sub-tree messages.

The loss rate analysis in Fig 6 indicates the gain of the Semantic Network Coding version. This result is very dependent on the activity inside the network. In our experiment, traffic reduction of Semantic Network Coding version has a good effect on transmission error rates. The router nodes keep a good ratio in both experiments, but some nodes far in the tree have difficulties receiving/transmitting data where there is heavy traffic load in the standard version (e.g. node 19).

V. Conclusion

Data collection in data-oriented IoT applications can lead to significant consumption of the yet limited resources of connected objects. Often, for a sensor, calculating a result right in place, costs less in terms of energy consumption than transmitting data. We saw that In IoT applications, priority is mainly given to the transmission of data to the Cloud, where it is then processed and reduced. However, processing the data in the device can provide a gain in network longevity at the cost of a loss of data accuracy, which can be accepted depending on the use case.

In this paper, we have presented an analysis to reduce data transmissions, by applying network coding techniques which focus on the semantics of the data to be transmitted. The experiment was conducted on the Contiki operating system which offers end-to-end IPv6 connectivity. Our approach significantly decreases power consumption within the network, limiting data forwarding at each hop for each node, especially for the most important ones for the sustainability of the network's existence. The possible evolution of this work consists in being able to dynamically adapt the network coding function used, or in proposing a passive network mode, triggering network coded measures only with a granularity adapted to the demand.

VI. Acknowlegment

This work was supported by the FUI SCORPION project (Grant no. 17/00464) and CNRS PRESS project (Grant no. 239953).

References

[1] Texas instruments cc2420 datasheet. https://www.ti.com/lit/ds/symlink/cc2420.pdf.

[2] Texas instruments cc2538 datasheet. https://www.ti.com/lit/ds/symlink/cc2538.pdf.

[3] E. Al-Hawri, N. Correia, and A. Barradas. Design of network coding based reliable sensor networks. *Ad Hoc Networks*, 91:101870, 2019.

[4] S. A. Alvi, F. ul Hassan, and A. N. Mian. On the energy efficiency and stability of rpl routing protocol. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1927–1932. IEEE, 2017.

[5] S. Balakrishna and M. Thirumaran. Semantics and clustering techniques for iot sensor data analysis: A comprehensive survey. In *Principles of Internet of Things (IoT) Ecosystem: Insight Paradigm*, pages 103–125. Springer, 2020.

[6] S. Cherrier, Y. Ghamri-Doudane, S. Lohier, and G. Roussel. Services Collaboration in Wireless Sensor and Actuator Networks: Orchestration versus Choreography. In *17th IEEE Symposium on Computers and Communications (ISCC'12)*, page 8 pp, Cappadocia, Turquie, July 2012.

[7] S. Cherrier and Y. M. Ghamri-Doudane. The "object-as-a-service" paradigm. In *Global Information Infrastructure and Networking Symposium (GIIS), 2014*, pages 1–7. IEEE, 2014.

[8] S. Cherrier, I. Salhi, Y. Ghamri-Doudane, S. Lohier, and P. Valembois. Bec3: Behaviour crowd centric composition for iot applications. *Mobile Networks and Applications*, pages 1–15, 2013.

[9] A. Dunkels, B. Gronvall, and T. Voigt. Contiki-a lightweight and flexible operating system for tiny networked sensors. local computer networks. In *Annual IEEE Conference on, 0*, pages 455–462, 2004.

[10] T. Li, W. Chen, Y. Tang, and H. Yan. A homomorphic network coding signature scheme for multiple sources and its application in iot. *Security and communication networks*, 2018, 2018.

[11] H. Nasiri, S. Nasehi, and M. Goudarzi. Evaluation of distributed stream processing frameworks for iot applications in smart cities. *Journal of Big Data*, 6(1):52, 2019.

[12] C. H. Oliveira, Y. Ghamri-Doudane, C. E. F. Brito, and S. Lohier. Optimal network coding-based in-network data storage and data retrieval for iot/wsns. In *2015 IEEE 14th International Symposium on Network Computing and Applications*, pages 208–215. IEEE, 2015.

[13] A. Palavalli, D. Karri, and S. Pasupuleti. Semantic internet of things. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, pages 91–95. IEEE, 2016.

[14] G. Peralta, R. G. Cid-Fuentes, J. Bilbao, and P. M. Crespo. Homomorphic encryption and network coding in iot architectures: Advantages and future challenges. *Electronics*, 8(8):827, 2019.

[15] F. Safara, A. Souri, T. Baker, I. Al Ridhawi, and M. Aloqaily. Prinergy: a priority-based energy-efficient routing method for iot systems. *The Journal of Supercomputing*, pages 1–18, 2020.

[16] Y. Sun, T.-Y. Wu, G. Zhao, and M. Guizani. Efficient rule engine for smart building systems. *IEEE Transactions on Computers*, 64(6):1658–1669, 2014.

[17] N. Tsiftes, J. Eriksson, and A. Dunkels. Low-power wireless ipv6 routing with contikirpl. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 406–407. ACM, 2010.

[18] F. Xhafa, B. Kilic, and P. Krause. Evaluation of iot stream processing at edge computing layer for semantic data enrichment. *Future Generation Computer Systems*, 105:730–736, 2020.