# Chained estimation for data reduction-driven routing in wireless sensor networks

Brandon Foubert, Christian Salim, Nathalie Mitton

# Chained estimation for data reduction-driven routing in wireless sensor networks

Brandon Foubert
*Univ. Grenoble Alpes*
*CNRS, Grenoble INP, LIG*
38000 Grenoble, France
brandon.foubert@univ-grenoble-alpes.fr

Christian Salim
*Junia*
*Computer Science and Mathematics*
59000, Lille, France
christian.salim@junia.com

Nathalie Mitton
*Inria*
59650 Villeneuve-d'Ascq, France
nathalie.mitton@inria.fr

***Abstract*—Wireless sensor networks are a precious tool for numerous use cases: studying the fauna and flora, measuring various metrics and so much more. Here we focus specifically on wireless multihop networks, in which the funneling effect is too often responsible of the unbalance of energy consumption, causing in turn nodes failures. In an attempt to mitigate the effects of funneling, the scientific literature proposes among others traffic reduction based on data estimation. The lead idea being that if two phenomena are correlated, then maybe we could estimate the way one goes based on the way the second one do. This effectively reduces the amount of data nodes may have to transmit while preserving the information. We propose in this article a similar method, but here not only limited to a pair of nodes. With our scheme we can reduce the traffic of several nodes over a whole multihop route of communications in a network. We have implemented that scheme and experimented with real hardware which shows good results.**

***Index Terms*—correlation, estimation, reduction, pearson, multihop**

## I. Introduction

A wireless sensor network is a set of constrained devices which communicate using wireless technologies. In such networks, each node is usually not able to send direct messages to each and every other node of the network. To bypass this limitation, nodes can send their messages to a neighboring node. In turn, the neighbor can forward the message to one of its own neighbors, and so on until it reaches its final destination. The data passes through hop to hop: we refer to this kind of networks as multihop networks. Nodes can try to reach different destinations in the network. Frequently, the nodes have to send data outside the wireless sensor network. However, usually few nodes have a connection to an outside network, *e.g.*, the Internet. Such a kind of nodes are called sinks and they collect a lot of data from the other nodes of the network. Communications converge towards the sinks: this is a well known traffic model called convergecast.

The data convergence caused by convergecast traffic is known as the funneling effect. A significant effect of funneling is that generally, the closer a node is located from a sink, the more data it has to forward from node located further from the sink. This induces an imbalance in the workload distribution. The energy consumption caused by data forwarding is significant. As the energy expense is unequal in the network, some nodes will run off of battery quicker than others. Furthermore, congestion causes loses and collisions, which in turn means resending messages, costing more energy, and so on and so forth. When it happens, and if there is no alternative to the route provided by the failed nodes, then the whole network may collapse. As an example, we can see in Figure 1 an illustration of the funneling effect in an extreme case. In the remainder of this article, $N_i$ will refer to node $i$, while $NBR(i)$ will refer to the set of nodes in the direct vicinity of $N_i$. In Figure 1, $N_A$ is the only node with an available link towards the base station. Thus, other nodes in the network have to forward all of their data to $N_A$. In turn $N_A$, forwards messages to the base station. Thereby, in addition to his own data, $N_A$ have to take care of the traffic from the whole network behind him. In such a case, $N_A$ will empty all of his available energy quickly. When the battery will no longer be able to sustain $N_A$ electrical needs, the other nodes of the network will not be able to send messages to the base station anymore.

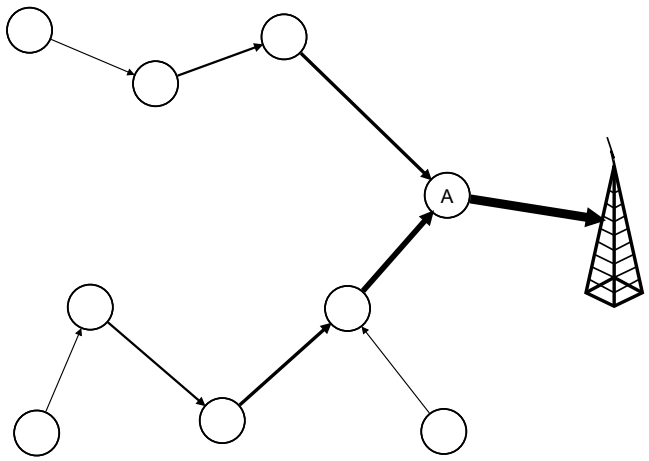


Fig. 1. Depiction of the funneling effect. $N_A$ is the network's cornerstone and forwards messages from the whole network.

The funneling effect is a well known issue of multihop wireless sensor networks. So common that it may very well have contributed to the recent popularity of single hop wireless technologies such as LoRa [1]. Such technologies simplify the network with a star topology, where each node is able to join

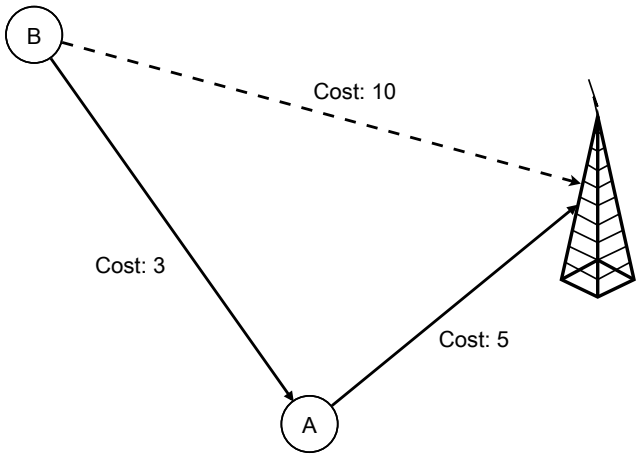


Fig. 2. Subpart of a previous experimental network where $N_B$ chose to forward his data to $N_A$ at his expense.

the sink in a single hop. However such technologies suffer from their own shortcomings. Thus in previous works [2], we proposed to use multiple technologies in a single network. In our work, nodes are able to chose between long range or multihop route depending on their specific needs. This effectively mitigate range and coverage issues, and takes into account the diversity of use cases in the network. However, in such a network, a node selects the route that is the most interesting according only to his own needs. Thus other nodes may sometimes suffer from an egoist choice of a node, as forwarding data increase the energy expense of nodes. This is something we've seen often in previous experiments. In Figure 2, we can observe a simplified example. Here $N_B$ has to chose between two routes to join the sink. The first one is a direct communication to the sink, while the second one pass through $N_A$. In this example, the second route is more interesting for $N_B$ since it has a lower cost. Thus $N_B$ is going to forward its data through $N_A$. However this is unfavorable for $N_A$ since it's going to increase its energy consumption.

Switching from a multihop to a long range technology cannot resolve every issues without creating new ones. To alleviate the nefarious effects of funneling, in this article we propose an original data reduction method based on chained estimations. Considering Figure 2, the main idea is that if data measured by $N_A$ and $N_B$ are correlated, then the sink could estimate $N_B$'s values based on $N_A$'s values. At the same time, $N_B$ computes the estimations the same way the sink does it. If estimations are accurate enough, $N_B$ does not need to send its own measurement anymore. This enables an effective traffic reduction. If estimations are too inaccurate compared to the real measurements, $N_B$ will be able to detect it. It can then simply send a message with the real measure, like it would have without the data reduction method. To that end, we leverage the Pearson correlation coefficient, which reflects how much datasets are correlated. Then, we detail an algorithm to use data reduction not only between a pair of nodes but for a whole multihop route. That scheme has been implemented and tested on real hardware to assess its efficiency.

The remainder of this article is structured as follows: Section II introduces works related to considered subject. Section III presents the Pearson correlation coefficient and the way we compute it. Section IV details the estimation method we use for our data reduction scheme. Section V explains the method we propose enabling data reduction over a entire multihop route. Section VI shows the experimental setting we used to assess the efficiency of our method and the obtained results. Finally, Section VII concludes this work.

## II. Related work

Data reduction can happen by trend determination, learning algorithms, and so on. In this section, we present some of the related works on data reduction in wireless sensor networks.

The authors of [3] propose a data reduction method based on the correlation of different variables measured by a single node. The variables' correlation is determined by the use of a Bayesian inference method. In the experiments described, it is used to send only the atmospheric pressure measurements, from which the estimations of the temperature are computed. This effectively reduce the amount of data sent by the nodes, but this does not take into account correlation between several nodes, nor does it propose a mechanism for reduction over several hops. In the article [4], authors describe a data reduction system based on two conjoint mechanisms. First, the amount of data sent by the nodes is adjusted automatically depending on the variance of the dataset. This is to say that only significant differences in the measurements are sent, while negligible ones are disregarded. Second, the sink estimate the measurements of the nodes. This way, nodes doesn't need to send each one of the collected data. Nodes compute the estimation in the same way to make sure it is accurate enough, otherwise a message containing the real value is sent. Such a system provides an effective data reduction, however is still doesn't provide a mechanism to deal with multiple hops over a route. From the works available in the literature we can see that our approach for multihop data reduction is unique.

Numerous method of correlation for data reduction are available in the literature, as we can see in [5]. However, in a similar way as for the works we mentioned in this section, there's no method of data reduction available in a multihop fashion. This is why we propose in this article an original method to enable multihop data reduction based on chained estimations. Our work is based on the Pearson correlation coefficient, which reflects the correlation of datasets. If a strong correlation is detected between data, nodes taking part in a multihop route are able, using our method, to greatly reduce the amount of data forwarded, and effectively reduce the nefarious impact of funneling in wireless sensor networks.

## III. Pearson correlation coefficient

The Pearson correlation coefficient is a statistical tool that reflects the existence or nonexistence of a linear correlation between two variables [6]. A coefficient comprised in the

interval $]0,1]$ means that variables are positively correlated, *i.e.*, both grow conjointly. On the contrary, a coefficient comprised in the interval $[-1,0[$ reflects a negative correlation, which roughly means that when one increases the other one decreases, and vice versa. A coefficient close to 0 means that the considered variables are not correlated.

Let's consider two real variables $X$ and $Y$. We refer to their Pearson correlation coefficient as $\rho_{XY}$. Its definition is the ratio between the covariance of $X$ and $Y$ and the product of their standard deviation. The covariance is the mean of the product of the deviation from the mean, and thus, for $n$ discrete values of $X$ and $Y$, the classic formula to compute $\rho XY$ is shown in (1).

$$\rho_{XY} = \frac{\sum_{i=1}^{n}(x_i - \overline{X})(y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{Y})^2}} \tag{1}$$

$$\rho_{XY} = \cos\alpha = \frac{\overrightarrow{X}.\overrightarrow{Y}}{\|\overrightarrow{X}\|.\|\overrightarrow{Y}\|} \tag{2}$$

Let's consider both our variables $X$ and $Y$ as vectors $\overrightarrow{X}$ and $\overrightarrow{Y}$ in an $n$ dimensions space. Then we can use a geometric interpretation of the Pearson correlation, where the coefficient $\rho XY$ equals the cosinus of the angle $\alpha$ between $\overrightarrow{X}$ and $\overrightarrow{Y}$. Based on the cosinus formula between two vectors, we obtain (2), which is equivalent to (1) for computing $\rho XY$.

$$P_{XY} = \rho_{XY}^2 \times 100 \tag{3}$$

Finally, let's define $P_{XY}$ the correlation percentage between $X$ and $Y$ computed as shown in (3).

Let's consider an example with two variables $TA$ and $TB$ which are the temperatures of two close locations. Vectors $\overrightarrow{TA}$ and $\overrightarrow{TB}$ both contains a subset of the measured temperatures. Let's consider those vectors to be like shown in (4).

$$\begin{aligned}\overrightarrow{TA} &= \{2,3,5,8,13,21\}\\ \overrightarrow{TB} &= \{2.8,4.8,7,12.8,18.2,33.6\}\end{aligned} \tag{4}$$

Based on (2), we get:

$$\rho_{TA\,TB} = \frac{1099.6}{2\sqrt{178}\sqrt{1703.92}} \approx 0.9983$$

Consequently, we have a correlation percentage $P_{TA\,TB} = 99\%$. This shows a very strong correlation between $TA$ and $TB$. In such a case, it may then very well be possible to estimate one as a function of the other. In the next section, we'll detail the method we use for computing such an estimation.

## IV. Estimation for data reduction

The Pearson correlation coefficient, computed such as shown in Section III, reflects the strength of the correlation between two variables. Considering wireless sensor networks, we consider here the variables to be phenomenons measured by two independent nodes. If those phenomenons are strongly correlated, then based on the Pearson coefficient and the set of measured data we are able to estimate the next measurements. As an example, for two monitored close locations, it's very likely that temperatures measurements from both nodes are going to be correlated. In the same idea, temperature and atmospheric pressure are also generally correlated. In this section, we detail our estimation method used for data reduction.

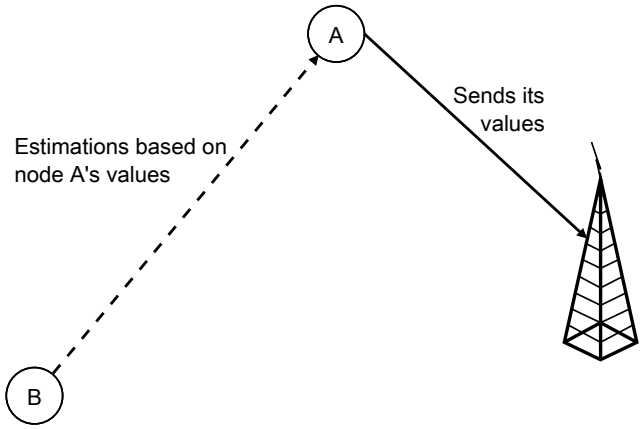


Fig. 3. Simple data reduction example with a pair of nodes. $N_B$'s measurements can be estimated based on $N_A$'s measurements.

Let $X$ and $Y$ be the datasets of measurements of two variables, both of size $n$. Based on the known data, we can compute the mean ratio between $X$ and $Y$ such as seen in (5). We refer to it as $R_{XY}$.

$$R_{XY} = \frac{\sum_{i=1}^{n}\frac{y_i}{x_i}}{n} \tag{5}$$

Based on the known values and the next value of set $X$ we can then estimate the next value added to the set $Y$. This is done following (6).

$$y_{n+1}^{\rho} = y_n + (x_{n+1} - x_n) \times R_{XY} \times \rho_{XY}^2 \tag{6}$$

Let's consider an example with the nodes from the simplified network we can see in Figure 3. Let's assume that $N_A$ has all the necessary parameters needed for the estimation of the next value of $N_B$, based on (6). In such a case, $N_B$ does not need to send its values to $N_A$, because the sink, as well as any equipment located further in the network and knowing the necessary values, are able to compute the next measurement of $N_B$ based on the new values sent by $N_A$. However those are estimations, which may be subject to inaccuracy. To make sure of the estimations' accuracy, we need to have a mechanism for $N_B$ to warn others that the estimation it computed does not match with the real measurement.

To that end, let's assume that $N_B$ has, as well as $N_A$, all parameters needed for computing the estimation of its own next measurement, based on (6). In such a case, $N_B$ is able to know exactly the same value computed by $N_A$. It is then able to compare its measurement with the computed estimation. If that estimation's accuracy isn't sufficient, or

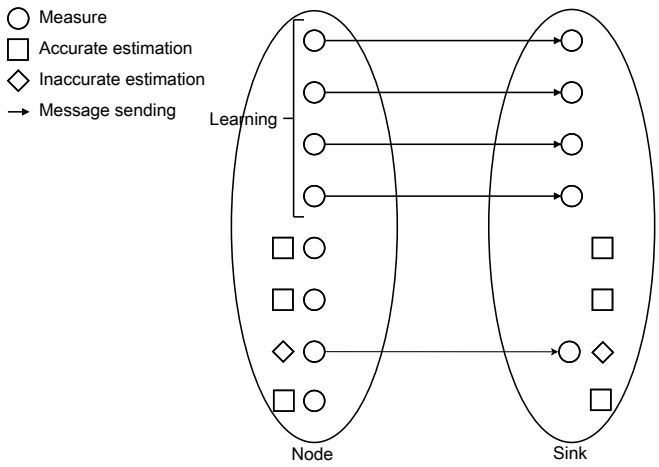


Fig. 4. Depiction of data reduction. The sink estimates the node's values based on the measurements it receives from another node. When there's an inaccuracy, the node sends a correction message.

just plainly wrong, $N_B$ can simply send a message to $N_A$ to both warn that the estimation is wrong as well as share the real measurement value. An illustration of this mechanism is depicted in Figure 4.

To be able to decide if an estimation is accurate enough or not, we need to set accuracy boundaries. To that end, let $B^-$ be the inferior threshold and $B^+$ be the superior threshold. Considering the data of $X$ and $Y$, let $y_{n+1}$ be the next real measurement. Then the estimation's accuracy is determined based on (7) and (8).

$$B^- = (\frac{1-\rho_{XY}}{2} + \rho_{XY}) \times y_{n+1} \tag{7}$$

$$B^+ = (\frac{1-\rho_{XY}}{2} + 1) \times y_{n+1} \tag{8}$$

Let $y^\rho_{n+1}$ be the next estimated value of set $Y$. It has to be like $B^- < y^\rho_{n+1} < B^+$. If not, it's mandatory to correct the wrong estimation with the sending of the real measurement. In such a case for example with a network like in Figure 3, $N_B$ would send the real value to $N_A$ which would forward it to the sink.

Let's consider again the examples we've seen in section III and in 3. Let's assume that $N_A$ and $N_B$ both measure the temperature in two distinct locations. $N_A$ is building $TA$ the set of its measurements, while $N_B$ does the same for the set $TB$. $N_A$ knows the whole set of values from $TB$, $N_B$ forwards its data through $N_A$. We need a mechanism so that, in the other way, $N_B$ knows all of the values from $TA$. To that end, we use a listening scheme similar to the one we used in our previous work for the dissemination of routing data dissemination [2]; *i.e.*, communication overhearing. Indeed, since $N_A$ sends its messages to the sink, and since $N_B$ is able to forward data through $N_A$, then $N_B$ is able to overhear $N_A$'s messages in an opportunistic way. Thus $N_B$ is able to know all of the values from the set $TA$.

Let $TA$ and $TB$ be like the vectors $\overrightarrow{TA}$ and $\overrightarrow{TB}$ at the end of section III, *i.e.*, like in (9).

$$\begin{aligned} TA &= \{2, 3, 5, 8, 13, 21\} \\ TB &= \{2.8, 4.8, 7, 12.8, 18.2, 33.6\} \end{aligned} \tag{9}$$

As seen in Section III, the correlation coefficient is such as:

$$\rho_{TATB} = \frac{1099.6}{2\sqrt{178}\sqrt{1703.92}} \approx 0.9983$$

Based on (5), we compute the sets' mean ratio such as:

$$R_{TATB} = 1.5$$

Let's now assume that the next measurement from $N_A$ is $TA_6 = 34$. $N_A$ can then estimate the next measurement of $N_B$ based on (6), which gives us:

$$TB^\rho_6 = TB_5 + (TA_6 - TA_5) \times R_{TATB} \times \rho^2_{TATB} \approx 53.0346$$

$N_B$ computes exactly the same value, and measures its next value which we assume to be $TB_6 = 53$. The thresholds computed by $N_B$ to ensure the $TB^\rho_6$'s accuracy are based on (7) and (8) and equals to:

$$\begin{aligned} B^- &\approx 52.9555 \\ B^+ &\approx 53.0445 \end{aligned}$$

In this case, we have $B^- < TB^\rho_6 < B^+$. The estimation is thus accurate enough, and $N_B$ doesn't need to emit anything for correction nor for sharing its real measurement. This enables $N_B$ to save up the energy it would have use to send $TB_6$, but also enables $N_A$ to save up the cost of receiving and forwarding $TB_6$. In the worst case, if $TB^\rho_6$ wasn't comprised between $B^-$ and $B^+$, then $N_B$ would just send a correction message to the sink through $N_A$ to share $TB_6$.

## V. Chained estimations for multihop reduction

The algorithm detailed in Section IV enables a data reduction scheme based on the estimation of correlated data. Ideally, we could just extend that for nodes over a route : the sink would collect data from the last node of the route. Then it could estimate the measurements of the next hop based on these data, and from those estimations estimate the next hop and so on and so forth. However it's not that simple since not only the estimation process needs numerous parameters, but also because the nodes of the route has to be able to detect and correct estimations inaccuracy when occurring. Setting up only a few tweaks, we nevertheless conceived an estimation scheme practical for a whole multihop route. This scheme allow for a fair distribution of the energy consumption over the route. In this section, we detail the algorithm we propose for enabling data reduction over a multihop route.

Let's now consider Figure 5, which essentially just adds $N_C$ in the network. Each node $N_A$, $N_B$ and $N_C$ has a set of measurements $TA$, $TB$ and $TC$. As explained in Section IV,

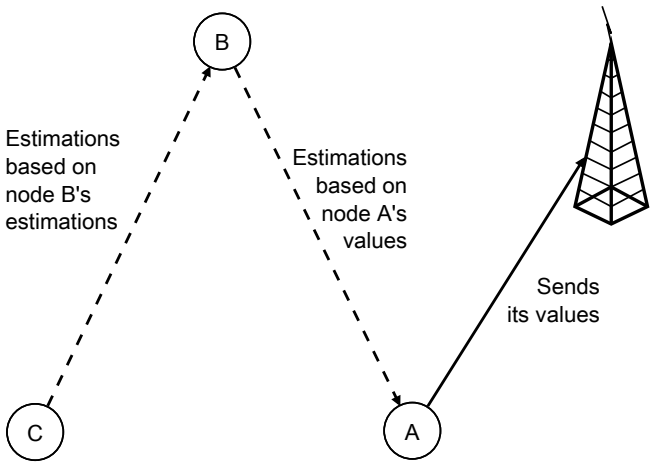


Fig. 5. Simple data reduction example over a route. $N_B$'s values are estimated based on the ones from $N_A$. $N_C$'s values are estimated from $N_B$'s values estimations.

$N_A$ can estimate measurements of $N_B$. In the same fashion, $N_B$ can compute the estimations of its own measurements, as it overhears the data of $TA$ $N_A$ sends to the sink. $N_B$ can then correct the inaccurate estimation by simply sending the real value to the sink through $N_A$ if needed.

That scheme of estimation needs time to initialize, and during that learning phase values are sent in a classic fashion, *i.e.*, $N_B$ sends its data to $N_A$ which in turn forwards it to the sink. When enough values are exchanged to detect a correlation the data reduction becomes effective. When effective, any equipment located further up in the network, *e.g.*, the sink, knows the values of $TA$ and $TB$, and thus those equipment can compute $\rho_{TA\,TB}$. This enables the data estimation of $N_B$, based on the method detailed in Section IV. However, for a device to be able to estimate measurements of $TC$, the actual scheme is not viable. $N_B$ is indeed capable of computing estimations of $N_C$ based on its own measurements estimations. But to make sure the accuracy of these estimations is enough, $N_C$ needs to do the same computations. For that it needs to know $N_B$'s values, however since $N_B$ is using data reduction, it does not send all of its data to $N_A$ anymore and thus no overhearing is possible.

To get around this issue, the answer we propose is that in a multihop route with data reduction active, each node sends to its direct neighbors the estimations it computes of its own values. Those messages are not intended to be forwarded to the sink, but only to be taken into account by the direct neighborhood of each node. This way, each node only has to assume the energy consumption of sending his own measurements (for the last node before the sink) or sending its own estimations to its neighborhood. Thus the accumulation issue caused by the funneling effect is effectively canceled. Take note that except the last node on the route before the sink, each and every node on the route sends estimations and not real measurements. Sending real measures would decrease the accumulation of inaccuracy caused by chained estimations. However devices located further up the network will estimate values based on chained estimations and thus nodes have to do it the same way to compare estimations with the real values.

Let's get through yet another example to clarify all of that. Considering 5, when data reduction is effective, the scheme we propose works like that: $N_A$ sends its real measurements to the sink. The sink, based on values from $TA$ and $TB$ collected during the initialization phase, is able to estimate the next measurements of $N_B$. $N_B$ on its side, overhears the values of $TA$, and compute the estimations of its own measurements, the same way the sink does it. If those are inaccurate, $N_B$ sends correction messages. The sink, based on values of $TB$ and $TC$ learned during the initialization phase, and based on the estimations of the next values of $TB$, is able to compute the estimation of the next values of $TC$. $N_B$ sends its own estimations of values of $TB$ to its direct neighborhood, and those are heard by $N_C$. This way, $N_C$ can compute the estimations of its own values from $TC$, based on the estimations of $TB$, *i.e.*, the same way the sink does it. If estimations are inaccurate, $N_C$ can simply send a correction message which would be forwarded through $N_B$ and $N_A$ towards the sink.

**Algorithm 1** : Chained estimation

**Require:** $N_i$ node $i$ ; $BR_i$ best route of $N_i$ ; $N_{i-1}$ next hop of $BR_i$ ; $S$ hops number of $BR_i$ ; $Ti = \{Ti_k | k = 1, 2, ..., j\}$ measurements dataset of $N_i$

```
for N_i do
  measure Ti_{j+1}
  if S = 1 then
    send Ti_{j+1} forwarded by BR_i
  else if S > 1 then
    compute B^-_{j+1} and B^+_{j+1} as a function of Ti_{j+1}
    if S = 2 then
      estimate Ti^ρ_{j+1} based on T(i-1)_{j+1}
    else if S > 2 then
      estimate Ti^ρ_{j+1} based on T(i-1)^ρ_{j+1}
    end if
    if B^-_{j+1} < Ti_{j+1} < B^+_{j+1} then
      send Ti^ρ_{j+1} addressed to NBR(i)
    else
      send Ti_{j+1} forwarded by BR_i
    end if
  end if
end for
```

In a general setting, let's consider a route from node $N_i$ until node $N_j$, which in turn relay data outside the network, *i.e.*, sending it to a sink. After the learning phase, data reduction becomes active. $N_j$ which is the last node before the sink still function in a classic fashion, *i.e.*, it sends its measurements to the sink and those are overheard by $NBR(j)$. The previous hop of the route $N_{j-1}$ compute its own estimations based on the values of $N_j$ overheard. It then sends those estimations to $NBR(j-1)$. The pattern repeats for every node until $N_i$. The algorithm we've explained is depicted formally in Algorithm 1.

With our method, as long as data are correlated, nodes forward less much messages compared to the classic fashion.

The strength of correlation shared between the nodes of a single route is a direct indication of the energy savings that a route can offer. Thus, its interesting to note its pretty simple to take this metric into account in the route selection process using a multicriteria routing protocol, *e.g.*, the one we conceived in previous work [2]. Indeed, our protocol is based on a multiattribute route selection method, which allows for any attribute to be taken into account in the process.

As an example, we can take the reduction rate offered by a route into account as an attribute of the decision matrix. For that, we can use either the mean Pearson correlation coefficient between the pair of nodes making up the route, or the mean reduction rate. For the second option, it would suffice to just compute the mean reduction rate between each pair of nodes. The reduction rate is a ratio between the number of measured values and the number of sent values. Let $TR_{ij}$ be the reduction rate for the pair of nodes $N_i$ and $N_j$. $TR_{ij}$ is computed like shown in (10).

$$TR_{ij} = (1 - \frac{number\ of\ sent\ values}{number\ of\ measured\ values}) \times 100 \quad (10)$$

## VI. Experimentation

To access the efficiency of our multihop data reduction scheme, we have implemented and experimented the method we proposed in this article on real hardware. The implementation was done with MicroPython on FiPy devices from Pycom. Such devices are based on ESP32 cores and provides five different wireless communications technologies. The Pycom company provides a firmware which includes a port of MicroPython to allow quick prototyping.

In our experiments, we used three datasets of 143 values each. Those contained temperature measurements, and 48 values were used for the learning phase. The topology used was like the one we can see in Figure 5, with the three nodes $N_A$, $N_B$ and $N_C$. For those three nodes, and the datasets used, we had the followings correlation coefficient between datasets: $[\rho_{TATB} \approx 0.9947, \rho_{TBTC} \approx 0.9918]$ and $[R_{TATB} \approx 1.0894, R_{TATB} \approx 0.5023]$. In terms of reduction rate, the obtained result for $N_B$ was a rate of $96\%$ and for $N_C$ a rate of $87\%$. It means that using our method, the traffic was reduced to respectively $4\%$ and $13\%$ of its original size.

However, from the difference of reduction rate between $N_B$ and $N_C$, we also observe that the estimations' accuracy decreases quickly as the nodes' depth in the route increases. This is due to the spread and build-up of the estimations' inaccuracies.

## VII. Conclusion and future works

Multihop networks allow nodes to forward data hop by hop. Such networks frequently have a convergecast scheme of traffic toward sinks. Thus communications converge and it causes a well known issue called the funneling effect. In turn funneling causes unequal energy consumption in the network as nodes closer to the sinks generally spend more energy.

To mitigate funneling, we proposed in this article a data reduction method over multihop route. Based on the Pearson correlation coefficient, our method enables to estimate the second set's values based on the first set's values. This way, nodes that measure strongly correlated variables don't need to send all of their collected data. Devices which know the necessary parameters are then able to compute data estimations based on the known values. To assess the estimations' accuracy, the nodes also compute the estimations of their own values. They can then compare values from their estimations to the real measurements. If those aren't accurate enough, nodes can simply send a correction message holding the real value.

In order to enable data reduction, not only between a pair of nodes, but for all nodes taking part in a multihop route, we propose a chained estimation scheme. Here, a device is then capable of computing chained estimation based on the measurements done by a single node, the one closest to the sink. At the same time, to ensure the accuracy of those estimations, nodes taking part in the route also compute the estimations of their own measurements. This only requires the estimations that the next hop compute and then share with its neighborhood. If the estimation is inaccurate, the node can simply send a correction message which will be relayed to the sink. That algorithm enables effective data reduction, which in turn distribute more fairly the workload and the energy cost of forwarding messages to the destination.

We assessed the efficiency of our method using real hardware, which showed an interesting reduction rate but also the build-up of inaccuracies caused by the chained estimations. For future works, we'd like to experiment further using bigger prototype networks, but also find a way to mitigate that inaccuracy accumulation.

## Acknowledgment

This work has been partially supported by the French Ministry of Research projects PERSYVAL-Lab under contract ANR-11-LABX-0025-01 and DiNS under contract ANR-19-CE25-0009-01.

## References

[1] L. Alliance, "A technical overview of lora and lorawan," *White Paper, November*, vol. 20, 2015.

[2] B. Foubert and N. Mitton, "RODENT: a flexible TOPSIS based routing protocol for multi-technology devices in wireless sensor networks," *ITU Journal on Future and Evolving Technologies*, vol. 2, no. 1, Apr. 2021. [Online]. Available: https://hal.inria.fr/hal-03165426

[3] C. Razafimandimby, V. Loscrì, A. Maria Vegni, D. Aourir, and A. Neri, "A Bayesian approach for an efficient data reduction in IoT," in *InterIoT 2017 - 3rd EAI International Conference on Interoperability in IoT*, Valencia, Spain, Nov. 2017. [Online]. Available: https://hal.inria.fr/hal-01620373

[4] G. Tayeb, A. Makhoul, D. Laiymani, and J. Demerjian, "A distributed real-time data prediction and adaptive sensing approach for wireless sensor networks," *Pervasive and Mobile Computing*, vol. 49, 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01948895

[5] G. Rajesh and A. Chaturvedi, "Correlation analysis and statistical characterization of heterogeneous sensor data in environmental sensor networks," *Computer Networks*, vol. 164, 2019.

[6] K. Pearson, "Note on Regression and Inheritance in the Case of Two Parents," *Proceedings of the Royal Society of London Series I*, vol. 58, Jan. 1895.