Chapter

WSN for Event Detection Applications: Deployment, Routing, and Data Mapping Using AI

Kamel Abbassi, Mohamed Hechmi Jeridi and Tahar Ezzedine

Abstract

In the 20th century, computers were senseless brains, but today, thanks to sensor networks, they can feel things for themselves. This major trend has given rise to many wireless sensor networks with the ability to sense the environment, deliver findings and process those data appropriately. Within this trend, this chapter outlines deployment and routing strategies as well as data handling practices. For convenience, the most encompassing application to consider is that of event detection.

Keywords: IoT, LLN, WSN, sensing, coverage, connectivity, deployment, routing, data fusion, machine learning

1. Introduction

Wireless sensor networks consist of many sensor nodes connected and providing valuable information for further processing to serve industry, people and society. The trend in WSNs is toward miniaturization, rapid deployment in large applications, reliable routing of data and its proper handling.

The broad description is based on four basic facts which are:

Capture: referring to the action of transforming an analog physical quantity into a digital signal. In our case, a study on the types of sensors seems necessary for the adequate choice for the application detection of events.

Concentrate and route: studying current routing techniques and refurbishing them to suit the event detection application.

Extract and Process: the data collected in their binary forms will be exploited through the use of AI techniques in perfect symbiosis with the proposed general architecture.

Store and Present: the fact of aggregating data, produced in real time, meta tagged, arriving in a predictable or unpredictable way, and the ability to restitute the information in a way that is understandable by Man, while offering him a means of acting and/or interacting.

2. Deployment issues

Identifying sensor node location and deployment strategies will be the first step toward building the network. Determination of sensor placement and rollout

method depends on several criteria, mainly the intended applications, surrounding area, estimated deployment time and cost.

2.1 Nodes deployment and connectivity

2.1.1 Strategies of nodes deployment

Wireless sensor nodes may be enormously affected by node deployment. In fact, Wireless nodes deployment is able to extend network lifetime, efficient reliability and routing, well preserving of network energy, ensuring connectivity, etc. Placing nodes in a defined zone of interest is often not pre-determined at the outset of network layout and deployment. Indeed, the manner of how to locate the sensor nodes depends mainly on the application type and the operating conditions surrounding them. In WSNs, both random and deterministic methods of node placement can be applied [1, 2].

It should be noted here that roll-out can occur all at once or can be a continuous process where additional nodes could be redeployed in the given zone, thus two deployment strategies can be identified in WSN: uniform distribution and non-uniform one.

2.1.2 WSN connectivity

Communication modeling in low-power and lossy networks is difficult because nodes transmit at reduced power and radio links lack sufficient reliability. Hence, this research adopts the disk model as a deterministic communication method to ease the process of analytical computation [3, 4]. The model considers each wireless node "n_i" as being efficient in acquiring and transmitting data to nearby nodes. The communication range marked " R_i " is a function of the level of transmission power of a node. Accordingly, two nodes "n_i" and "n_j" can communicate with one another bilaterally only if the Euclidean distance there between is at least the minimum of their communication range (Eq. (1)).

$$d(n_i, n_j) \leq \min\{R_i, R_j\} \tag{1}$$

2.1.3 Detection in lattice WSNs

A network of wireless sensors is deployed in an area of interest for event detection, so each point in the area is covered by a sensor node if the Euclidean distance between the sensor node and this point is less than the sensing range R_S.

The probability that a sensor detects an event in the area of interest at a distance "d" is given by Eq. 2.

$$P(d) = \begin{cases} \mathbf{1} & if \;\; d \leq R_S \\ \mathbf{0} & if \;\; d > R_S \end{cases} \tag{2}$$

Due to the environmental and geographical conditions relative to the area of interest and manufacturers' requirements, we will define an uncertainty parameter "e" to make the model more realistic. Hence, the probability that a sensor will detect an event in the area of interest at a distance "d" is given by Eq. 3.

$$\boldsymbol{P(d)} = \begin{cases} \mathbf{1} & if \;\; \mathrm{d} < (\mathrm{R_S} + \mathrm{e}) \\ \boldsymbol{e^{-\alpha(R-R_S)^{\beta}}} & if \;\; (\mathrm{R_S} - \mathrm{e}) \leq \mathrm{d} \leq (\mathrm{R_S} + \mathrm{e}) \\ \mathbf{0} & if \;\; \mathrm{d} > (\mathrm{R_S} + \mathrm{e}) \end{cases} \tag{3}$$

α and β are the peripheral oriented parameters relative to the sensor node.

2.2 How detection and connectivity issues impact WSN coverage?

In deterministic deployment, sensor nodes are arranged around preset positions and carefully selected locations. Therefore, geometric shape designs can be adopted to form the coverage area as shown in **Figure 1**.

The firs purpose is to identify the appropriate number of sensor nodes to be deployed for full connectivity in the area in question.

Let "A" be the size of the AOI and "As" the area related to the sensor node's geometric configuration. The area of the geometric patterns of the sensor nodes is determined by knowing the length of the sides, designated by $\mathbf{l_x}$, i.e. $\mathbf{l_c}$ for the quadrilateral shape and $\mathbf{l_t}$ for the triangular one, and $\mathbf{l_h}$ for the hexagonal structure. Then, the estimated number of sensor nodes "N," is given by Eq. 4.

$$N = \frac{A}{A_S} \tag{4}$$

To have full connectivity, the pattern surface for each sensor node is given as follows:

$$As(\boldsymbol{con}) = \begin{cases} lc^2 = Rc^2 & \text{quadrilateral shape} \\ \frac{\sqrt{3}}{2} lt^2 = \frac{\sqrt{3}}{2} Rc^2 & \textit{triangular shape} \\ \frac{3\sqrt{3}}{4} lh^2 = \frac{3\sqrt{3}}{4} Rc^2 & \textit{hexagonal shape} \end{cases} \tag{5}$$

The number of sensor nodes adequate to ensure total connectivity is given by:

$$N(\boldsymbol{con}) = \begin{cases} \frac{A}{Rc^2} & \text{quadrilateral shape} \\ \frac{A}{\frac{\sqrt{3}}{2} Rc^2} & \textit{triangular shape} \\ \frac{A}{\frac{3\sqrt{3}}{4} Rc^2} & \textit{hexagonal shape} \end{cases} \tag{6}$$

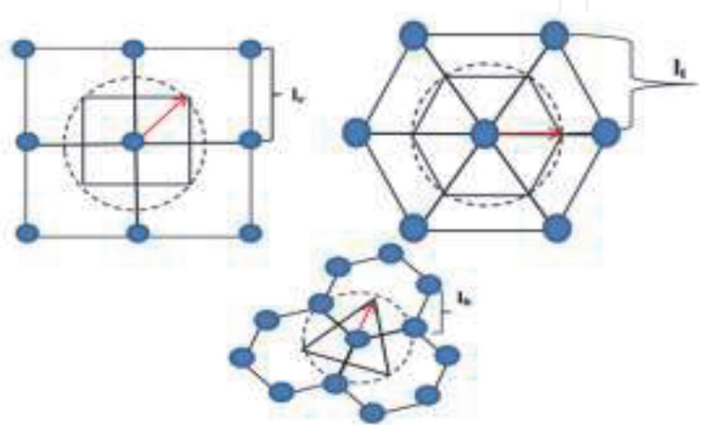

Figure 1.
Geometrical shapes of cover areas.

The second purpose is to identify the appropriate number of sensor nodes to be deployed to ensure total coverage; the geometric pattern surface relative to a sensor node is given by:

$$As(\boldsymbol{cov}) = \begin{cases} Rs^2 & \text{quadrilateral shape} \\ 3\dfrac{\sqrt{3}}{2}Rs^2 & \textit{triangular shape} \\ \dfrac{3\sqrt{3}}{4}Rs^2 & \textit{hexagonal shape} \end{cases} \quad (7)$$

The number of sensor nodes adequate to ensure total detection coverage is given by:

$$\boldsymbol{N}(\text{cov}) = \begin{cases} \dfrac{\boldsymbol{A}}{\boldsymbol{Rs^2}} & \textbf{quadrilateral shape} \\ \dfrac{\boldsymbol{A}}{\dfrac{3\sqrt{3}}{2}\boldsymbol{Rs^2}} & \textbf{\textit{triangular shape}} \\ \dfrac{\boldsymbol{A}}{\dfrac{3\sqrt{3}}{4}\boldsymbol{Rs^2}} & \textbf{\textit{hexagonal shape}} \end{cases} \quad (8)$$

In harsh environments such as a battlefield or a disaster region, deterministic deployment of sensors is very risky and/or infeasible. In this case, random deployment often becomes the only option.

Providing connectivity within the network is a fundamental objective in wireless sensor networks for communication and data exchange between sensor nodes. A sensor is supposed to be connected if and only if it has a direct or indirect (multi-hop) communication path to a destination. For the sake of brevity, the explanation deals only with the case of a direct communication (a single jump) denoted 1-connected.

A wireless sensor network is said to be connected if all sensors are connected and if there is no isolated sensor in the network. Indeed, the number of its neighbors in its communication range R_C is called the "degree of sensor node." It is in this sense that we will say that a sensor is isolated if its degree is equal to zero.

Consider arbitrarily a sensor in the network, the probability that it is isolated is equivalent to the probability that there is no neighboring sensor in its communication radius R_c, expressed in Eq. 9.

$$P_{iso}(S_i) = e^{-P\pi Rc^2} \quad (9)$$

The probability that the sensor S_i is not isolated is given by:

$$P_{non-iso}(S_i) = 1 - e^{-P\pi Rc^2} \quad (10)$$

If we consider that all sensors are deployed independently and uniformly in the area of interest, then they have the same probability of being non-isolated. Thus, the probability that there is no isolated sensor is as follows (Eq. 11)

$$P_{non-iso} = \prod_{i=1}^{N} P_{non-iso}(S_i) = \left(1 - e^{-P\pi Rc^2}\right)^N \quad (11)$$

The case where there is no isolated sensor in the network is required but not enough to say that this one stays connected. This gives us just the upper limit of the

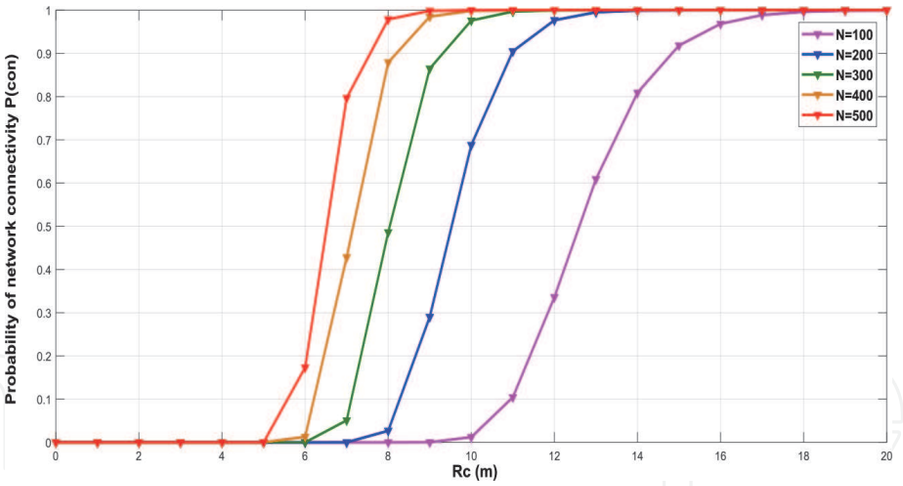

Figure 2.
Probability of network connectivity.

network connectivity. Also, the sensors need to have at least K neighbors in order to have K degree of connectivity in the network. In other words, a network in which all the sensors have at least one neighbor (i.e. there is no isolated sensor) implies that it is highly connected (high probability level of connection).

For more practical purposes we can consider a 100 m x 100 m area, changing the number of deployed nodes (from N = 100 to N = 500) and the coverage radius RC (from 0 m up to 20 m). Looking at the probability of network connectivity we can see that: there is a crucial communication radius over which a high probability of network connectivity exists. This radius is determined based on the configuration of the sensor nodes in accordance with the communication range as shown in **Figure 2**.

In this first section, we looked at the fundamental issues of wireless sensor network deployment, namely coverage and connectivity [5]. The first reflects the ability of the sensor network to provide detection in the area of interest and the second reflects the reliability with which the information collected by the sensor nodes will be transmitted for processing. But it is clear that the routing of data itself deserves to be properly considered for an appropriate sensor network dedicated to its intended application.

3. Toward proper routing

Routing data across nodes toward endpoints becomes the next logical priority, once deployment scenarios are dealt with.

Commonly, collected sensor data should be directed to a sink node via multi-hop routing. Certain applications may have different requirements, or constraints on the quality of the data transmission, such as end-to-end delay, jitter or churn, packet loss, etc. A set of constraints must always be respected. In fact, there are several proactive and reactive routing protocols that have been proposed for WSN [6]. One of them is the standardized proactive RPL (Routing Protocol for Low-Power and Lossy Networks), designed basically for “many to one” communications. A Destination Oriented Acyclic Graph (DODAG) is used for data collection, which is mainly a tree directed to the sink node. The calculation of the DODAG tree parameters is based on an objective function that is defined according to a single measurement. This section considers the use of RPL in WSN.

3.1 Purpose

Classically, self-powered, low-speed wireless personal area networks, such as 802.15.4, were considered unable to support IP. Now, 6LoWPAN and RPL made this possible. Even with the careful design of RPL routing, there is a need to assess this protocol in the context of more real-world applications.

In this section, focusing on a concrete application, that of event detection, we will assess the impact sending control packets at different time intervals, using different numbers of nodes and different transmission ranges. We will show how the variation of the control packet sending intervals can affect the performance of the routing protocol for low power lossy networks, considering the packet delivery rate and the power consumption.

3.2 Impact of network environment on the RPL

3.2.1 Preliminary details

RPL forms a Destination Oriented Directed Acyclic Graph (DODAG) (**Figure 3**), where each node has a rank calculated according to the cost metric and has a Preferred Parent (PP) as a gateway to the root [7]. In addition, RPL is regulated by three control messages ICMPv6 (Internet Control Message Protocol version 6) as defined in RFC 4443, namely:

- DODAG Information Object (DIO) which lets a node to discover an RPL instance,
- Destination Advertisement Object (DAO) allowing the propagation of destination information along the DODAG as well as the updating of the routing table on reception,
- DODAG Information Solicitation (DIS) for nodes that request to join the topology or to obtain more recent configuration information.

As for the objective function, it is indeed a mechanism enabling selection of the parent node by a child node, according to a metric of the DODAG tree. The main RPL specification does not have a built-in objective function, but both the zero-objective function (OF0) and the minimum order objective function with hysteresis (MRHOF) are recognized as defaulted OFs in this protocol [8, 9].

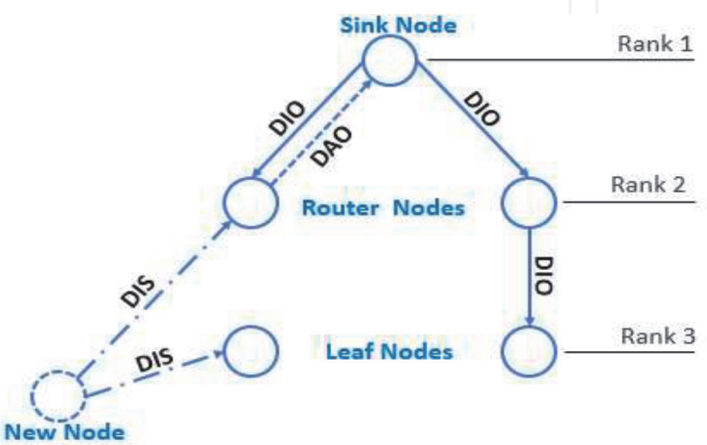

Figure 3.
DODAG Construction.

Proceeding with the actual data traffic handling arrangements, the analysis of the objective functions OF0 and MRHOF should be performed first. Indeed, OF0 is the Objective Function zero, which specifies how nodes select and optimize routes based on the minimum hop count to reach the parent node, while MRHOF stands for Minimum Rank with Hysteresis Objective Function which selects routes minimizing metrics, and uses hysteresis to reduce balancing in response to small changes in the metric.

To note that MRHOF works with Expected Transmission Count metrics (ETX) that are additive along a route and using the minimum rate to select the parent node and calculated according to Eq. (12).

$$ETX = \frac{1}{D_f \times D_r} \tag{12}$$

D_f denotes "forward delivery ratio" and indicates the likelihood of receiving the packet at the neighboring node.

D_r stands for "reverse delivery ratio" which calculates the probability of receiving the packet ack on receiver node side.

3.2.2 Procedures, tests, and evaluation

All simulations are carried out in the CONTIKI OS operating system using the Cooja simulator, considering the settings listed in **Table 1**. To perform the required simulations at various sending intervals (3 seconds, 5 s, 10s, 30s, 60s, 120 s, 300 s), we have modified the default parameter "PERIOD" in the "collect-common.c" file. Seeing that MRHOF is the default objective function for Contiki, we have modified the files rpl-conf.h and rpl-make. We had performed about 70 simulations in Cooja to evaluate the performance of the RPL objective functions in two scenarios.

Scenario 1: we fixed the network densities and modified the values of the TX range.

Scenario 2: we fixed the values of the TX range and modified the network densities.

In both scenarios, we have modified the sending intervals, allowing us to simulate highly constrained applications in lower SI values and normal applications in higher SI values.

In addition, a random topology will be used as this is the most suitable type chosen for the application. We implement a P2M topology, which means that there is only one sink node and the others are emitters, with a simulation time of about 10 minutes (600 seconds). The simulated platform was Tmote Sky under Unit Disk Graph Medium (UDGM) as the radio model. It should be noted that the Tmote Sky platform based on the MSP430 microcontroller with 10 KB of RAM and an

Settings	**Value**
Propagation model	UDGM with distance loss
Mote type	Tmote Sky
TX ratio	100%
Simulation time	600 seconds
Node position	Random

Table 1.
Used settings in the cooja simulator.

IEEE802.15.4 compatible radio is a constraint in the simulation of large networks because no more than 40 nodes can be simulated with reference to the hardware specification which leads to the limitation of the memory size of the routing tables [10]. The present work consists in testing the behavior of the two OFs in the RPL routing protocol by varying the SI sending intervals under various TX transmission ranges and different network densities. Following [11] the results of the simulation in the Cooja simulator could give the same results as in a practical experiment despite the external factors. Therefore, we will consider the Cooja results as practical tests. We collect all results obtained during all simulations and express them in graphs. We will discuss these results in subsections dealing only with each scenario.

Scenario 1: The network densities were set at 10 nodes and the Tx range values were varied from 50 m (**Figures 4** and 5) to 100 m (**Figures 6** and 7). It is clear that the longer the sending intervals, the lower the power consumption is in the case of OF0 and MRHOF, due to the fact that nodes consume more power since they send packets regularly and remain active in shorter intervals. Furthermore, Results indicate that MRHOF is more efficient than OF0 in terms of power consumption for one simple reason: MRHOF uses the ETX metric to build the routing table that reduces the amount of power used to transmit data between nodes.

The same conclusion for PDR, while increasing the sending intervals, the PDR value increases In fact, at tighter time intervals, nodes send packets repeatedly, leading to more load on delivered packets. Therefore, many packets do not reach destination because their capacity becomes exhausted. Thus, extending the send interval will increase the received packets, directly impacting the PDR value. Furthermore, we have noticed that when SI = 30s, MRHOF reaches 100% PDR. This could prove the idea that MRHOF performs better than OF0 in the PDR results because it reaches the full PDR ratio in fewer sending intervals.

Scenario 2: This scenario considers the TX range values at 100 m and changes the network densities from 10 nodes (**Figures 6** and 7) to 15 nodes (**Figures 8** and **9**). Increasing the transmission intervals reduces power consumption. However, power consumption for OF0 and MRHOF are too close in this scenario, but MRHOF is still better.

We also notice that by increasing the network density, the power consumption increases compared to scenario 1, but the difference is not very important because we also increase the transmission range, so that more nodes reach the sink node

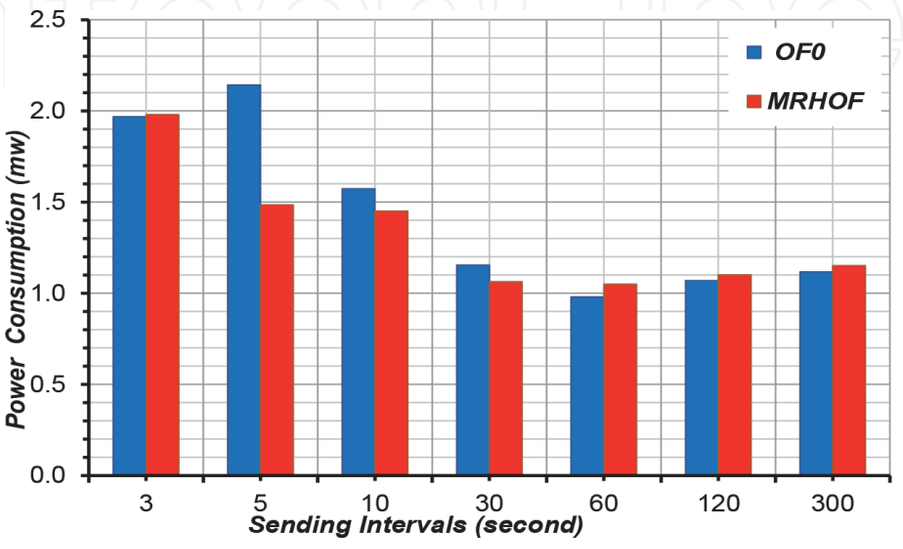

Figure 4.
Performance in power consumption (10 nodes, Tx = 50 m).

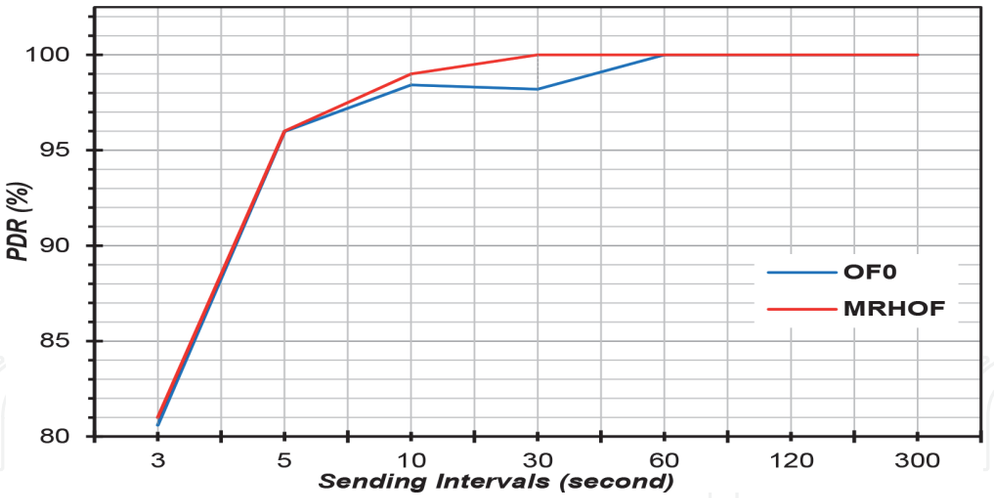

Figure 5.
Performance in PDR (10 nodes, Tx = 50 m).

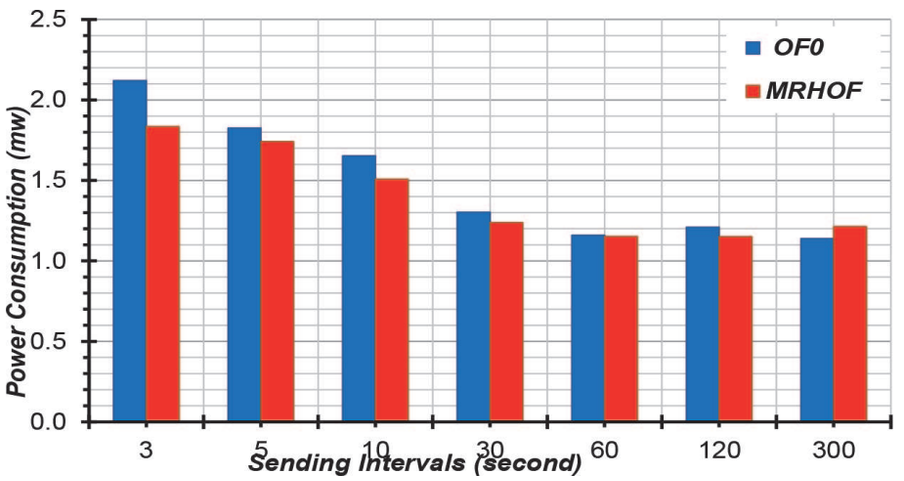

Figure 6.
Performance in power consumption (10 nodes, Tx = 100 m).

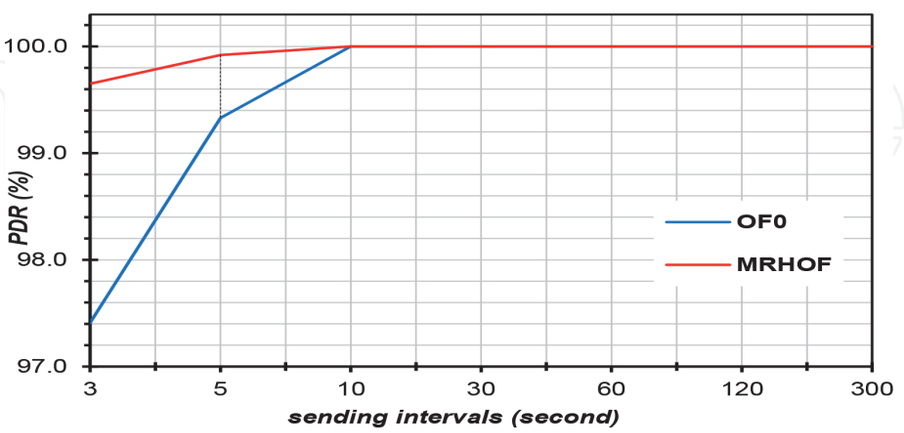

Figure 7.
Performance in PDR (10 nodes, Tx = 100 m).

without the need to switch from the transmitter to the receiver, thus lowering the power consumption level.

MRHOF reaches a higher PDR level in the former sending intervals, but OF0 is still not very performing and declined in 30 seconds of sending intervals due to

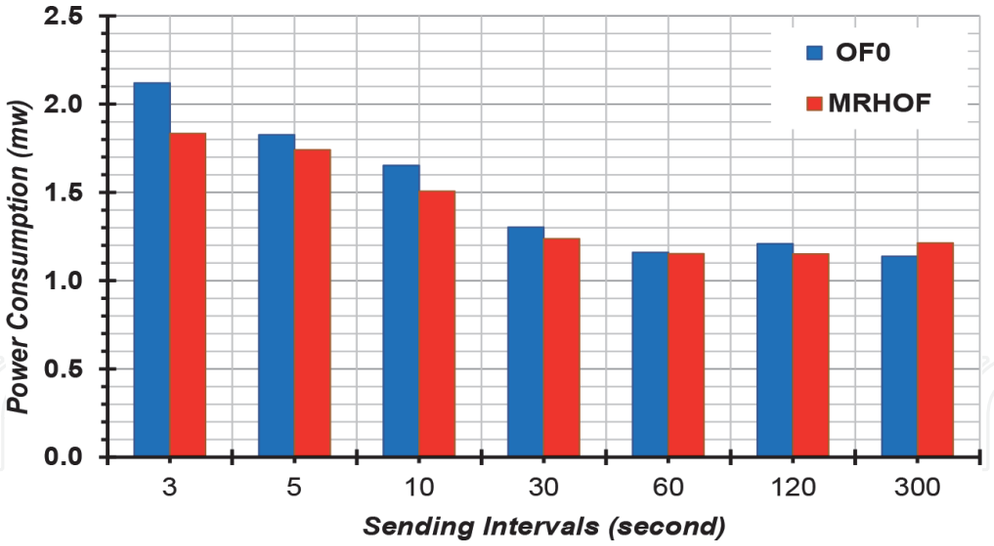

Figure 8.
Performance in power consumption (15 nodes, Tx = 100 m).

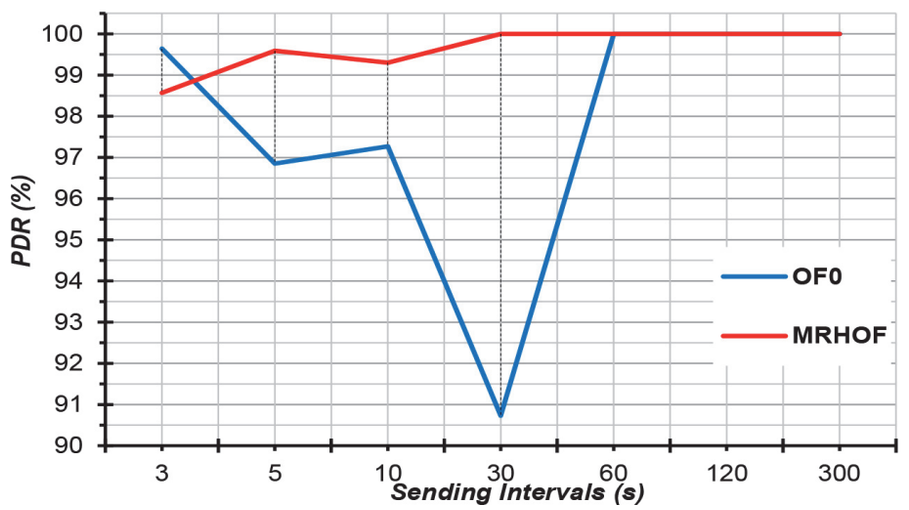

Figure 9.
Performance in PDR (15 nodes, Tx = 100 m).

inconsistent network structure, moreover not all nodes in this scenario were able to reach the destination node, what can be considered as unintended findings.

When we take the 60s as the value for the sending intervals, both MRHOF and OF0 had a 100% ratio for the PDR, confirming the fact that the PDR value increases as the sending intervals are incremented.

Finally, selecting an appropriate routing protocol to a given application like event detection is often important and necessary, however, it remains to be said that after the routing of data through the network, it is essential to process them correctly to benefit as much as possible from this content while keeping in mind the limitations of low capacity, low power and lossy networks. The following section outlines in more detail the proposed method for data treatment and mining.

4. Data querying and mining

Having carefully considered the deployment principles and the proper routing on the network, it remains to specify how to properly handle the data, most notably,

if one considers WSN as a database. For the purpose of the event detection applications currently in use the processing of requests is a big challenge, so it is necessary to optimize the execution time and the resources used.

To reduce the cost of executing aggregated queries in event detection applications, events are often pre-calculated and materialized. These aggregated views are commonly known as summary tables that can be used to assist in responding to requests containing information beyond the view it represents. Yet, this provides a significant gain in performance. Many optimization techniques are available in the published work, such as indexing and fragmentation [12–14]. In this section, we consider materialized views.

4.1 Materialized view

The materialized views technique is mainly used to reduce the workload execution time [15]. It is made of several joints considered expensive to compute and store, especially for large sensor networks. It prepares these joints in the form of materialized views (MV) so that they do not repeat themselves each time. There are several algorithms to create the VMs such as backpack, similarity between queries, cache, etc. We will deal with the Jaccard similarity index approach [16]. The proposed technique as presented in **Figure 10** consists in:

- The extraction of the workload to create the query attribute table (each query is presented by a vector).
- Calculation of the similarity between each pair of vectors.
- For each similarity group we create a cluster (only clusters with high similarity will be kept).
- A view will be created for each cluster.

4.1.1 Modelization

The Workload Q is formed by n queries $\{Q_1, \ldots Q_n\}$. A query is composed by k attributes $\{a_1, \ldots a_k\}$, each query is $Q_i = \left\{a_j^i\right\}, \forall i : 1..n, j : 1..k$.

The activation function used in this work is:

$$f(a_j) = \begin{cases} 1 * a_j \; if \; a_j \; used \; in \; Q \\ 0 * a_j \; else \end{cases} \tag{13}$$

The workload can be presented in form of matrix:

$$MatA = \begin{pmatrix} a_1^1 * f(a_1) & \cdots & a_k^1 * f(a_k) \\ \vdots & \ddots & \vdots \\ a_1^n * f(a_1) & \cdots & a_k^n * f(a_k) \end{pmatrix} \tag{14}$$

The database management system has only one Final Solution FS (guarantees the re-joining to all workload with minimal cost). FS is formed by a set of materialized views v. Each view is composed of one or more attributes. Based on, N attributes, we can find $2^N - 1$ views and the FS has a view between 1 and $2^N - 1$.

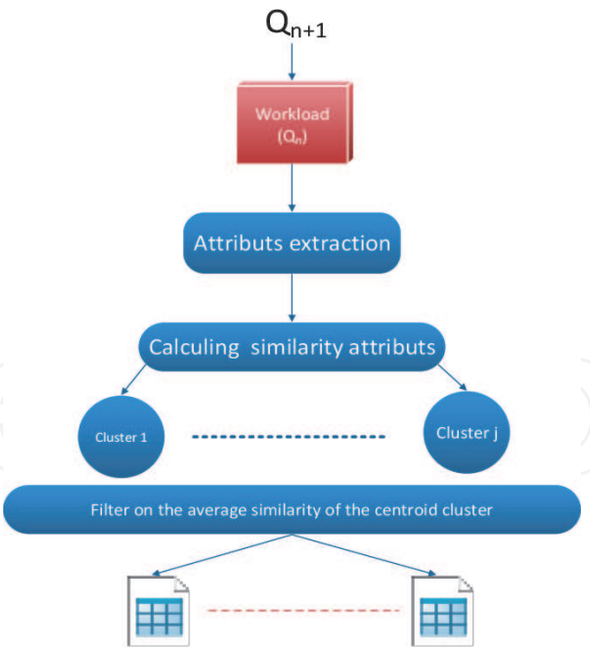

Figure 10.
Process of MV making.

$$FS^f = \{v_e^f\}, where e \in (1..2^N - 1),\ f = \left(1..2^{2^N-1} - 1\right) \tag{15}$$

For example, FS^1 composed by $v_1, v_3, and\ v_4$:

$$FS^1 = \{v_1^1, v_3^1, v_4^1\} \tag{16}$$

In order to verify if this materialized view v_e is included in the solution FS^f or no, the function $h(v_e)$ having the following form should be used

$$h(v_e) = \begin{cases} 1, & v_e\ if\ v_e\ used\ in\ FS \\ 0, & v_e\ else \end{cases} \tag{17}$$

The maximum number of final solutions is $(2^N - 1)^{2^N-1}$.
The final solutions are presented as follows

$$FS = \begin{pmatrix} v_1^1 * h(v_1) & \cdots & v_{2^N-1}^{(2^N-1)^{2^N-1}} * h(v_{2^N-1}) \\ \vdots & \ddots & \vdots \\ v_1^n * h(v_1) & \cdots & v_{2^N-1}^{(2^N-1)^{2^N-1}} * h(v_{2^N-1}) \end{pmatrix} \tag{18}$$

The references of the final solutions are stored in a vector VS (19).

$$VS = \{S_f\}, where\ f = \left(1..2^{2^N-1} - 1\right) \tag{19}$$

The sizes based on the final solutions are illustrated in **Figure 11**, where:
The first level illustrates all the attributes $\{a_1, .., a_n\}$ in the database tables.
The second level represents all the possible pointers of the materialized views i.e., $\{v_1, .., v_{2^n-1}\}$.

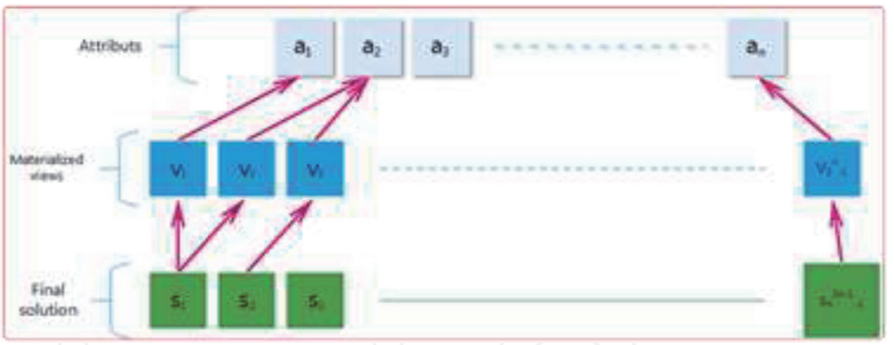

Figure 11.
Final solutions tree.

The third level presents all the possible final solutions where *FS* can be formed by one or all the materialized views. Thus, the maximum number of final solutions is $2^{2^n-1}-1$.

4.1.2 Learning practice

Our learning system consists of an input layer serves to receive the *MatA* matrix, an output layer to receive the *FS* matrix, and hidden layers (**Figure 12**). The $\{MatA,\ FS\}$ present the learning set. For each *MatA* input, the learning network gives a *FS* as a response.

The network is presented by the Eq. (20).

$$\mathrm{F} = \mathrm{G}(\mathrm{MatA}, \mathrm{W}), \textit{where } W \textit{ is the input weights} \tag{20}$$

Learning is defined by the process used to find $\hat{W}$ for all MatA such as $FS \cong \hat{FS}$, where FS and $\hat{FS}$ represent respectively, the final solution and the desired weight. Thus, the error function $Err(MatA, FS, W)$ (21) measuring the Euclidean distance between the MatA and output *FS* must be minimized.

$$Err(MatA, FS, w) = \|MatA - FS\|^2 = \sum_{i=1}^{n} (MatA_i - FS_i)^2 \tag{21}$$

The relationship between query-attributes Matrix and Final solutions matrix is given by Eqs. 22 and 23.

$$MatA * W + \beta = FS \tag{22}$$

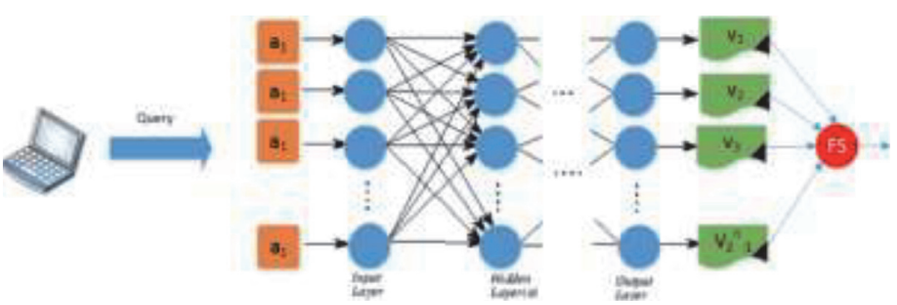

Figure 12.
Artificial neural network to create MVs.

Otherwise:

$$\begin{pmatrix} a_1^1 * f(a_1) & \cdots & a_k^1 * f(a_k) \\ \vdots & \ddots & \vdots \\ a_1^n * f(a_1) & \cdots & a_k^n * f(a_k) \end{pmatrix} * \begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix} + \beta$$
$$= \begin{pmatrix} v_1^1 * h(v_1) & \cdots & v_{2^N-1}^{(2^N-1)^{2^N-1}} * h(v_{2^N-1}) \\ \vdots & \ddots & \vdots \\ v_1^n * h(v_1) & \cdots & v_{2^N-1}^{(2^N-1)^{2^N-1}} * h(v_{2^N-1}) \end{pmatrix} \quad (23)$$

Note: β is the bias.

4.2 Algorithms

The proposed approach is structured in two algorithms, the one for generating VMs and the other for their prediction.

Algorithm 1: ***Building VMs***

1 ***Input****: Q: Workload, LP: learning period, Aused: Attributes used in Q, A: Attributes set*
2 ***Output****: FS: Final solution*
**/ Final solution = materialized Views set */*
3 ***Setup****:*
4 A: array () array of pointers of all attributes defined in the database
5 V: array () of pointers of all possible materialized views
6 FS: array () of pointers of all possible final solutions
7 ***Start***
8 ***While*** *(counter < LP) {*
9 Aused ← extraction(Q)
10 MatQA ← query_attributes(Aused,Q)
11 MatJS ← Jaccard_similarity (MatQA)
12 Graphs ← graph (MatJS)
13 Views ← cluster (Graphs)
14 Training_views (MatA, FS)
15 Counter ++
16 }
17 FS ← prediction_views(Q)
18 ***End***

Prior to the start, the training period must be tested (**Figure 13**). Once it is running, the algorithm follows the usual processes by creating the attributes of the query matrix and finding similarities between each vector.

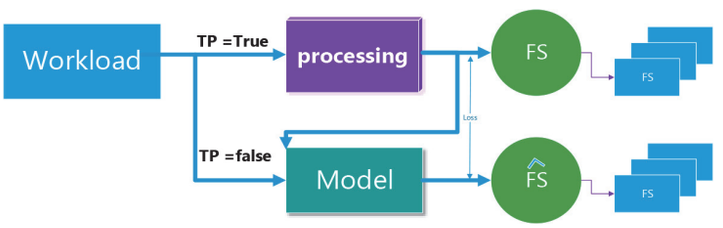

Figure 13.
Switching between the training phase and the prediction phase.

Algorithm 2: ***Predicting VMs.***

1 ***Input****: Qn + 1: next query, W: weight given by learning algorithm*
2 ***Output****: Final solution (Materialized Views set)*
3 ***Begin***
4 A = Extraction (Qn + 1)
*5 Views = A*W*
6 Return Views
7 ***End***

The complete process of creating MVs with the learning machine is illustrated in **Figure 14**.

First, all attributes used in the database tables are identified then every possible solution with the corresponding MV will be generated. Here only references for views are generated, the real MVs can be created in the learning step using Jaccard similarity technique. We identify the MVs per solution within this learning phase so as to find their optimal resolution. The neural network considers the attributes and views obtained when building the model. Once this learning step done, the optimal solution with the attributes present in the Qn + 1 workload will be found by the model.

4.3 Test and result

Concrete case examples will be considered to make the tests and deduce the results.

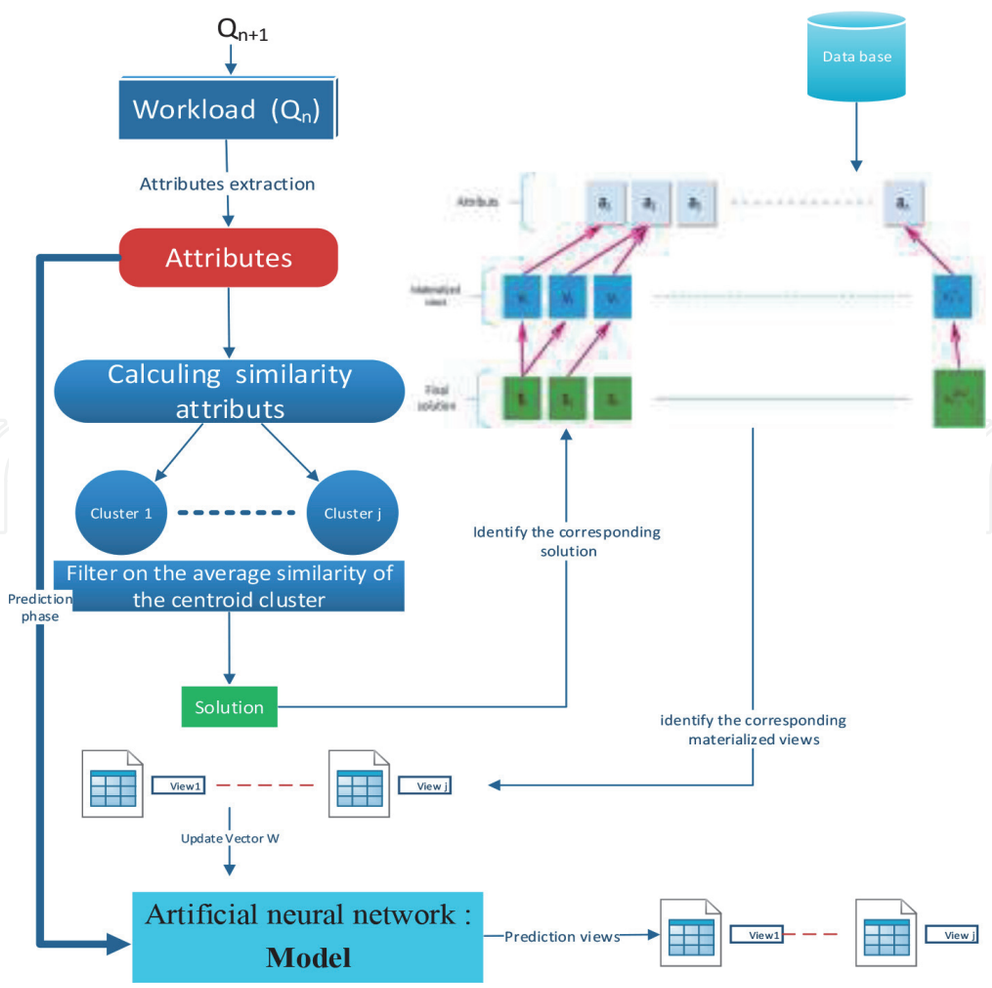

Figure 14.
Global architecture.

Initialization:

Considering a workload composed of seven queries $\{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7\}$ and five attributes $\{a_1, a_2, a_3, a_4, a_5\}$.

To proceed, store all the attributes in a list (**Figure 15**).

Next, a table composed of all possible MVs gets formed (**Figure 16**).

References set of materialized views

Then, all the references for the possible final solutions (PFS) will be created (**Figure 17**). In our test example, PFS = $2^{2^5-1} - 1 = 2147483647$.

Training phase:

Without being too sophisticated, three times {T1, T2 and T3} will be used in the example and we will measure the input weights W then we will present the experimental results of our approach

- **At time T1**, with one query $Q = \{Q_1\}$

	Q_1
a_1	1
a_2	1
a_3	0
a_4	1
a_5	1

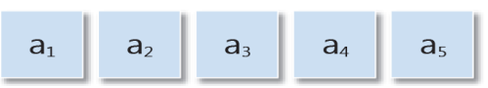

Figure 15.
Example of attributes set.

v_1	a_1	v_5	a_5	v_9	$a_1 a_5$	v_{13}	$a_3 a_4$	v_{17}	$a_1 a_2 a_4$	v_{21}	$a_1 a_4 a_5$	v_{25}	$a_3 a_4 a_5$	v_{29}	$a_1 a_2 a_4 a_5$
v_2	a_2	v_6	$a_1 a_2$	v_{10}	$a_2 a_3$	v_{14}	$a_3 a_5$	v_{18}	$a_1 a_2 a_5$	v_{22}	$a_2 a_3 a_4$	v_{26}	$a_1 a_2 a_3 a_4$	v_{30}	$a_1 a_3 a_4 a_5$
v_3	a_3	v_7	$a_1 a_3$	v_{10}	$a_2 a_4$	v_{15}	$a_4 a_5$	v_{19}	$a_1 a_3 a_4$	v_{23}	$a_2 a_3 a_5$	v_{27}	$a_1 a_2 a_3 a_5$	v_{31}	$a_1 a_2 a_3 a_4 a_5$
v_4	a_4	v_8	$a_1 a_4$	v_{12}	$a_2 a_5$	v_{16}	$a_1 a_2 a_3$	v_{20}	$a_1 a_3 a_5$	v_{24}	$a_2 a_4 a_5$	v_{28}	$a_2 a_3 a_4 a_5$		

Figure 16.
References set of materialized views.

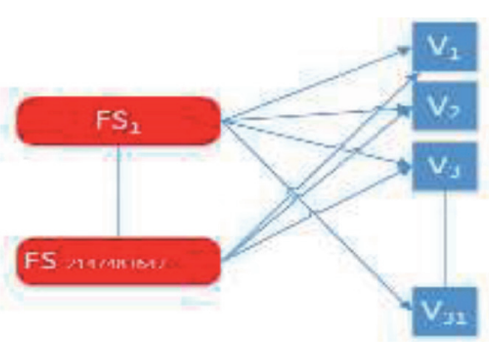

Figure 17.
Final solution references.

We need to set a matrix showing the similarity between two attributes with Jaccard Index *JI*. For example, $a_1 = (1\,1\,1\,1\,1\,1\,0)$ and $a_2 = (1\,1\,1\,1\,1\,0\,1)$ is computing as follow:

$$J(a_1, a_2) = \frac{a_1 \cap a_2}{a_1 \cup a_2} \tag{24}$$

J (a1, a2) = J (a1, a4) = J (a1, a4) = J (a2, a4) = J (a2, a5) =1, J (a1, a3) = J (a2, a3) = 0.

Only the pairs of attributes having a similarity > = 0.5 will be considered. The attributes having a similarity equal to 1 will be ignored. We can divide the attributes into 1 graph $C_1 = \{a_1, a_2, a_3, a_4, a_5\}$. The average similarity value for each centroid cluster is computed as follow: $(C_i)_{avg} = \frac{\sum_{j=1}^{2^n} J(a_i, a_j)}{2^n}$ where i ≠ j, $J(a_i, a_j)$ in Jaccard index between $a_i and a_j$. If the average similarity value is more than 0.5, the graph is considered a cluster. In this case, (C1) avg. = 1, {a1, a2, a4, a5} is a set of attributes exist in V29, where the final solution is FS29.

Then, finding $W = (w_1, w_2, w_3, w_4, w_5)$ with the following equation

$$1 * w_1 + 1 * w_2 + 0 * w_3 + 1 * w_4 + 1 * w_5 = 29 \tag{25}$$

	Q_1	Q_2
a_1	1	1
a_2	1	1
a_3	0	1
a_4	1	1
a_5	1	1

- **At time T2** with two queries: $Q = \{Q_1, Q_2\}$

J (a1, a3) = 0; J(a2, a3) = J(a3, a4) = J (a3, a5) = 0.5;
J(a1, a2) = J (a1, a4) = J (a1, a5) = J (a2, a4) = J (a2, a5) = J (a4, a5) = 1;
Graph C1 = {a2, a3, a5};
Graph C2 = {a1, a2, a4, a5}
(C1)avg = {J(a2, a3) + J(a2, a5) + J(a3, a5) }/3 = (0.5 +1+ 0.5) / 3
= 0,666667;
(C2)avg ={J(a1, a2) + J(a1, a4) + J(a1, a5) + J(a2, a4) + J(a2, a5) + J(a4, a5)}/6= 1;
{a2, a3, a5} ⇔ V23;
{a1, a2, a4, a5} ⇔ V29.
According to (17), we have:
FS1 ➔ V1 ... FS31 ➔ V31; FS32 ➔ V1 V2 ... FS61 ➔ V1 V31; FS91 ➔ V3 V4 ... FS118 ➔ V3 V31; ... FS466 ➔ V23 V29
So, we calculate $W = (w_1, w_2, w_3, w_4, w_5)$ with the following equation

$$2 * w_1 + 2 * w_2 + 1 * w_3 + 2 * w_4 + 2 * w_5 = 466 \tag{26}$$

	Q1	Q2	Q3
a1	1	1	1
a2	1	1	1
a3	0	1	1

a4	1	1	0
a5	1	1	0

- **At time t3** $Q = \{Q_1, Q_2, Q_3\}$

J (a3, a4) =J (a3, a5) =1/3; J (a1, a2) = J (a4, a5) =1
J (a1, a3) = J (a1, a4) = J (a1, a5) = J (a2, a3) = J (a2, a4)
=J (a2, a5) =2/3;
Graph C1 = {a1, a2, a4, a5} where similarity = 1
Graph C2 = {a1, a2, a3, a4, a5} where similarity = 1
(C1) avg. = (J (a1, a2) + J (a1, a4) + J (a1, a5) + J (a2, a4) + J (a2,a5) + J (a4, a5))/6 = 1;
(C2) avg. = (J (a1, a2) + J (a1, a4) + J (a1, a5) + J (a2, a3) + J (a2, a4) + J (a2, a5) + J (a3, a4) + J (a3, a5) + J (a4, a5))/9 = 6/9.
{a1, a2, a4, a5} ⇔V30;
{a1, a2, a3, a4, a5} ⇔V31
V30 and V31 exist in final solution FS496
We calculate $W = (w_1, w_2, w_3, w_4, w_5)$ *with the following equation*

$$3 * w_1 + 3 * w_2 + 2 * w_3 + 2 * w_4 + 2 * w_5 = 496 \tag{27}$$

5. Result

The host computer of the simulations runs under Linux Ubuntu 16.04 and has an Intel(R) xenon(R) CPU E5-2640 v4 processor running 2.4 GHz × 2.4 GHz (2processors) with 128 GB of memory. All the algorithms were implemented in Python 3. After a learning period, the experimental results (**Figure 18**) show a reasonable decreasing between the real solution and the prediction.

The part data mapping using AI was the final part of a quite complete process, beginning with sensors placement followed by the data routing till their smooth handling where we used the business intelligence to create materialized views in a dynamic processing. A mathematic approach has been proposed to present the workload and the attributes used. Next, we introduced our proposed methodology, split into two algorithms. The first is a neural network used for learning using

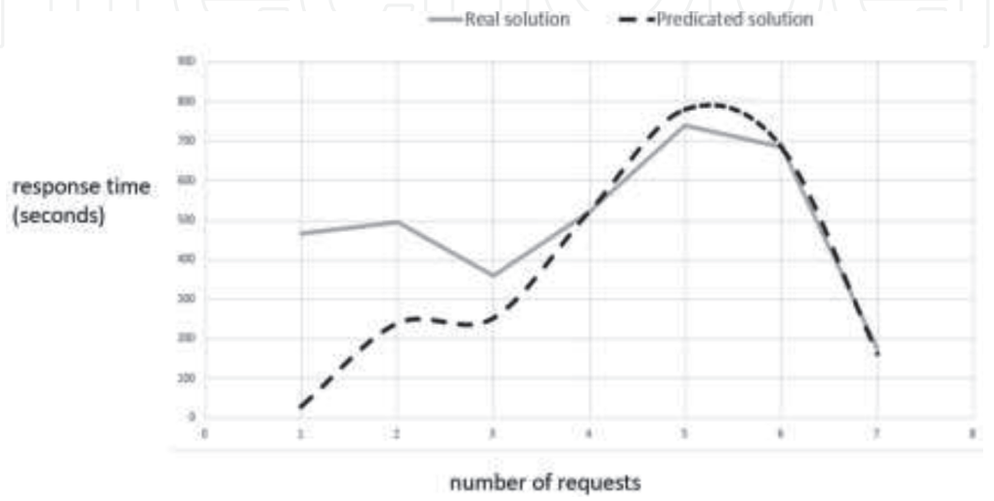

Figure 18.
Real solution vs. predicted solution.

similarity between attributes to provide a model to be used a second time. For the second algorithm, the network model is used to predict the future solution.

6. Conclusions

A wireless sensor network is a network of distributed autonomous devices that can cooperatively detect or monitor physical or environmental conditions. WSNs are used in many applications such as environmental monitoring, habitat monitoring, natural disaster prediction and detection, medical monitoring, and structural health monitoring, which we have encompassed in this work through the application event detection or fact detection. WSNs consist of a large number of small, inexpensive, disposable, autonomous sensor nodes that are typically deployed on an ad hoc basis over large geographic areas for remote operations. Sensor nodes are severely limited in terms of storage resources, computing capacity, communication bandwidth and power supply, which is why the decision was made to perform heavy computing outside the WSN network in computing centers. Indeed, the sensor nodes are grouped into clusters, and each cluster has a node that acts as the cluster head. All nodes transmit their sensor data to the cluster head, which in turn routes it to a specialized processing node(s) via multi-hop routing. In event detection applications, nodes detect the environment and immediately evaluate the data to determine its usefulness. If useful data (an event) is detected, the data is transmitted to the base station(s). Data traffic is difficult to predict events typically occur randomly and the resulting data traffic is sporadic. However, an amount of data must be exchanged for route management and network controls, even if no events are detected. A wireless sensor network, can, contrastingly, be considered as an intelligent, scalable system for monitoring and sensing the physical world's properties. In the last few years, the database research and development community claimed that if one considers a WSN as a database (which means that an essential feature of its smart design is the ability to perform explicit requests), this could lead to a considerable savings in the development of software engineering that runs a data acquisition program for the wireless sensor network, as well as a very favorable cost-effectiveness ratio due to the optimization of data queries. This work outlines a query computing model for the wireless sensor network that meets many of the requirements of considering the WSN as a database.

Acknowledgements

This work is part of a Tunisian-South African cooperation scientific research project. In this context, we thank the Ministry of Higher Education and Scientific Research of Tunisia. Our partner in the project, Distinguished Professor Qing-Guo WANG (https://www.scopus.com/authid/detail.uri?authorId=7408170159) acknowledges the financial support of the National Research Foundation of South Africa (Grant Numbers: 113340, 120106), which partially funded his research on this project.

Author details

Kamel Abbassi, Mohamed Hechmi Jeridi* and Tahar Ezzedine
National Engineering School of Tunis (ENIT), B.P. 37 1002, Belvedere, Tunis, Tunisia

*Address all correspondence to: mohamedhechmi.jeridi@enit.utm.tn

References

[1] Priyadarshi, Rahul, Bharat Gupta, and Amulya Anurag. "Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues." The Journal of Supercomputing (2020): 1–41.

[2] Banoori, Farhad, et al. "Deployment Techniques of Nodes in WSN and Survey on their performance Analysis." 2018 International Conference on Advanced Control, Automation and Artificial Intelligence (ACAAI 2018). Atlantis Press, 2018.

[3] Zhou, Baofeng, and Mehmet C. Vuran. "Network Time Connectivity for Wireless Networks." 2020 International Conference on Computing, Networking and Communications (ICNC). IEEE, 2020.

[4] Maheshwari, Aastha, and Narottam Chand. "A survey on wireless sensor networks coverage problems." Proceedings of 2nd International Conference on Communication, Computing and Networking. Springer, Singapore, 2019.

[5] Prakash, A., & Ansari, M. Y. (2019, October). Effective-coverage and connectivity in WSN. In Communication and Computing Systems: Proceedings of the 2nd International Conference on Communication and Computing Systems (ICCCS 2018), December 1–2, 2018, Gurgaon, India (p. 144). CRC Press.

[6] Snigdh, I., & Gosain, D. (2016). Analysis of scalability for routing protocols in wireless sensor networks. Optik, 127(5), 2535–2538.

[7] Tian, Hongliang, et al. "QoI-aware DODAG construction in RPL-based event detection wireless sensor networks." Journal of Sensors 2017 (2017).

[8] Solapure, Sharwari S., and Harish H. Kenchannavar. "Design and analysis of RPL objective functions using variant routing metrics for IoT applications." WIRELESS NETWORKS (2020).

[9] Onwuegbuzie, Innocent Uzougbo, Shukor Abd Razak, and Ismail Fauzi Isnin. "Control Messages Overhead Impact on Destination Oriented Directed Acyclic Graph—A Wireless Sensor Networks Objective Functions Performance Comparison." Journal of Computational and Theoretical Nanoscience 17.2–3 (2020): 1227–1235.

[10] Sha, Mo, et al. "Empirical study and enhancements of industrial wireless sensor–actuator network protocols." IEEE Internet of Things Journal 4.3 (2017): 696–704.

[11] Ai, Zheng-Yang, Yu-Tong Zhou, and Fei Song. "A smart collaborative routing protocol for reliable data diffusion in IoT scenarios." Sensors 18.6 (2018): 1926.

[12] Sebaa, Abderrazak, and Abdelkamel Tari. "Query optimization in cloud environments: challenges, taxonomy, and techniques." The Journal of Supercomputing 75.8 (2019): 5420–5450.

[13] Wu, Xingbo, Fan Ni, and Song Jiang. "Wormhole: A fast ordered index for in-memory data management." Proceedings of the Fourteenth EuroSys Conference 2019. 2019.

[14] Luo, Chen, and Michael J. Carey. "LSM-based storage techniques: a survey." The VLDB Journal 29.1 (2020): 393–418.

[15] Sebaa, Abderrazak, Amina Nouicer, and Abdelkamel Tari. "Impact of technology evolution on the materialized views: current issues and future trends." International Journal of

Business Information Systems 30.4 (2019): 427–462.

[16] D. C. D. Shrivastava, «selection and maintenance of materialized view and its application for fast query processing: A survey,» Proceedings of international journal of computer science and engineering survey, pp. vol.1, no 2, November 2010.