Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2021, 8(8):15-29**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Clusters Construction Mechanism for Junction Linear Wireless Sensor Networks (2CMJ)

Abdourakhmane Fall, Moussa Dethié Sarr, Cheikh Sarr

Computer Science Department, Université Iba Der Thiam de Thies, BP 967 Thiés, Senegal
Email: [abdourakhmane.fall, mdsarr, csarr]@univ-thies.sn

Abstract A Linear Wireless Sensor Network (LWSN) is a collection of sensors geographically arranged in a linear topology. These types of networks are applied today in several fields, their applications mainly concern areas based on linear form environments i.e. road or rail infrastructure border, waterway, oil, pipeline surveillance, gas etc.

There are two categories of LWSN topologies: Strictly linear networks with a single line starting from the sink, and linear networks with junctions, in which we have several lines with crossings called junctions.

In order to better organize the network, several techniques for forming the network in clusters have been proposed. These self-construction cluster algorithms organize the network into several groups of nodes called a cluster. The majority of these techniques, studied in the literature, are not adapted to a linear topology. The few exceptions to this rule become ineffective on linear topologies with junction areas.

In this paper, we propose, 2CMJ, a self-training cluster mechanism for wireless sensor networks with junctions. This solution is an extension of the M2CRL mechanism, taking into account the junction zones in the process of forming clusters. A mathematical model based on set theory has also been proposed in order to determine some characteristics relating to nodes, such as their Cartesian position with respect to the Sink.

2CMJ was evaluated on the Castalia / Omnet ++ simulator. The results obtained show a perfect formation of clusters, with zero orphan nodes and zero singleton clusters, with construction time much lower than the reference time.

Keywords Linear Wireless Sensor Networks, Linear Wireless Sensor Networks with Junction, Clustering algorithm, automatic topology construction

Introduction

Wireless Sensors Networks (WSNs) are a collection of sensors arranged in a specific topology, whose purpose is to collect, process, and transmit data for a specific application. WSNs can be deployed in a variety of environmental areas including volcanoes [7], forests, rivers, bridges [12], railways, commercial, military, medical and other applications, and are also widely used in industry, including production lines. These various and varied fields of application mainly define the topology according to which the WSN will be laid out, the network generally takes the form of the environment in which it is deployed. Thus there are several types of topologies for WSNs.

Linear Wireless Sensor Network (LWSNs) are a set of sensors arranged in a shape that can be represented with linear curves. LWSNs are currently used in a variety of applications: monitoring of bridges, road, rail, gas, oil and water pipelines [3-4] etc.

WSNs are subject to several constraints, the most important of which is undoubtedly its very limited energy reserves. Several solutions have been proposed in order to increase the service life of the sensors, and therefore of the entire network. One of the methods used consists of dividing WSN into several groups of sensors called

clusters [10]. Each cluster has a leader called Cluster Head (CH) whose role, among other things, is to collect the data sent by the nodes belonging to the cluster.

WSNs are no exception to the constraints affecting WSNs in general. Therefore, some methods used on other topologies, to overcome these constraints, can also be used in linear topologies.

Several clustered self-construction algorithms for WSNs have been proposed in the literature [6]. The majority of these algorithms concern networks with non-linear topologies. In [1] A Fall et al presents, M2CRL, a self-construction mechanism specific to linear wireless sensors networks. The latter solves the problem of the non-adaptability of old algorithms to linear topologies. However, the major constraint of this solution is that it is not applicable on a linear topology with junction areas.

In this paper we present 2CMJ, an extension of M2RCL [1] taking into account linear topologies with junction areas of varying complexity.

The paper will be organized as follows. In the first part 1, we will talk about the state of the art concerning LWSN. Part 2 will be devoted to the mechanism of cluster construction for networks with strictly linear topology M2CRL. In the third part 3, we present our solution as well as some applications of some properties of the set theory that would be applicable in a wireless sensor network with linear topologies. In the fourth part 4, we will present the results of the simulations performed. Finally, the last part 5 will present the conclusion and some perspectives.

1. State of the art

LWSNs are a special case of WSN, in which the sensors are deployed in a linear form. In this section we present a state of the art focusing on the different topologies for linear networks and on clustering techniques for LWSNs.

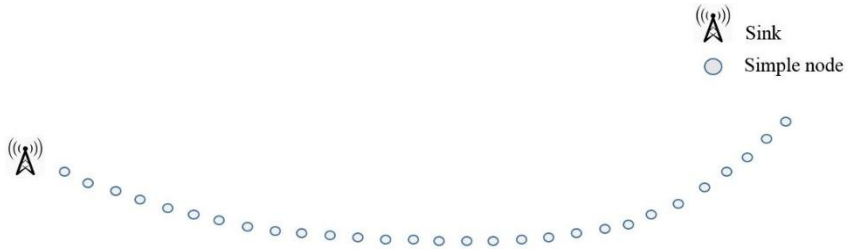
1.1 Topologies in LWSNs

LWSNs are essentially divided into two categories Figure 1: First we have networks with strictly linear topologies. In this type of network we note the existence of a single linear line starting from the sink. We also note a total absence of junction areas. These types of LWSNs can be deployed in environments such as bridges, border areas, certain roads or railways that do not have junction segments. Secondly, we have networks with linear topologies with junction areas. In these types of networks, we note the existence of several linear lines, some of which form crossing points called junctions, these junctions present several levels of complexity. The complexity of a junction is measured by the number of linear lines that emerge from the junction areas Figure 2.

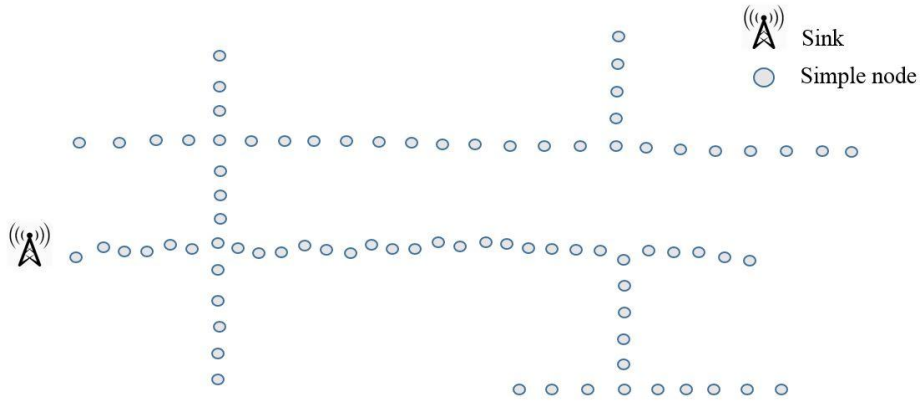
Depending on the placement of the sensor nodes and the role played by each sensor, we distinguish different types of topology in the literature

1.1.1 Topologies k-redundant

Redundancy, for a sensor, is given by its number of neighbours (within transmission range) in one direction. Therefore, the more neighbours the node has, the greater its redundancy.



(A): WSN: strictly linear



(B): WSN: with junction area

Figure 1: Topologies

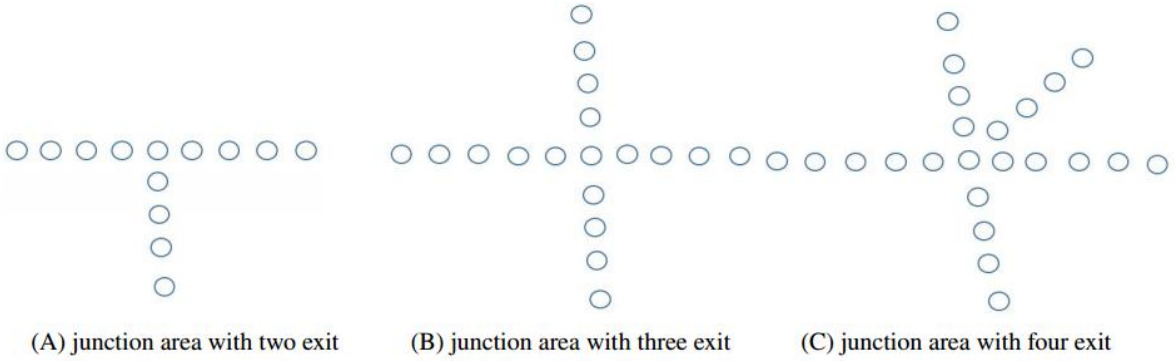


Figure 2: Junction area with varied complexities

In [8] the authors present a topology based on this redundancy concept. In this architecture a node with k neighbours in the direction of the sink, the same number of neighbours, also in the opposite direction, is referred to as a k-redundant network Figure 3.

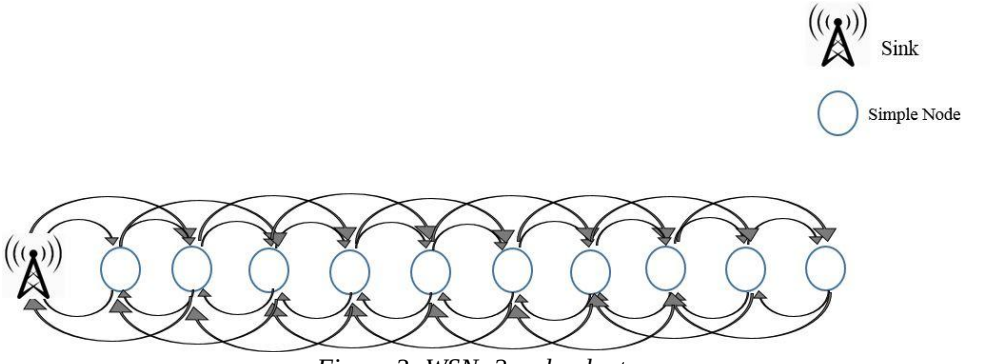


Figure 3: WSN: 2-redundant

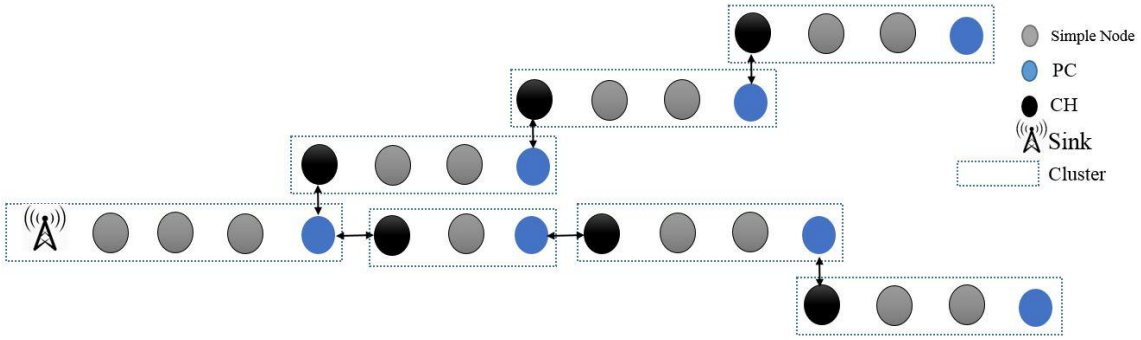


Figure 4: WSN: Long Thin Wireless Sensor Networks

1.1.2 Hierarchical topologies

A hierarchical topology is a topology in which the sensors have different levels of functionality. In [9] a three-level hierarchical topology is proposed, in this type of topology we have three types of nodes: Simple Sensors (NCS) which only collect the data to be transmitted, the (NCS) then transmit the collected data to Relay Nodes (NRD) whose role is to collect the data emitted by the (NCS). Once collected, the (NRDs) aggregate the data received and transmit the aggregated packets to Data Routing Nodes, which will route the data to Sink.

1.2 Clustering in LWSNs

In cluster topologies, nodes are formed into groups called clusters. In each cluster, one node is designated as the Cluster Head (CH). The CH, among other tasks, coordinates intra-cluster actions, can aggregate data issued by child nodes belonging to thesame cluster. This technique of logical network organisation makes a significant contribution to saving the residual energy of the sensors and thus increasing the lifetime of the entire network.

A large number of clustering algorithms have been proposed in the literature [10-11]. In [10], the authors have made a detailed and comparative study of most of the clustering algorithms proposed in the literature for both homogeneous and heterogeneous WSNs. In [11], the authors make a classification of the different algorithms studied, their classification is based on criteria such as data aggregation, energy consumption of nodes, CH rotation system, equalization of cluster size, CH centrality etc...

Most of these clustering algorithms are based on non-linear topologies. This parameter is not to be neglected insofar as the unsuitability of these algorithms for linear topologies makes them inapplicable to the latter. It was therefore necessary to think about clustering mechanisms specific to linear networks. These algorithms will have to take into account the fundamental aspect of LWSNs, i.e. linearity.

In [2] the authors propose a three-tier architecture based on two types of topologies: a non-linear topology and a linear topology. On the non-linear part, the LEACH [14] algorithm is used to create clusters. The Head clusters transmit their packets to the nodes located in the linear part, which in turn forward the packets to the base station.

In [13] the authors propose an addressing and routing mechanism based on a linear topology in a cluster tree. In the topology they propose, clusters are composed of simple nodes (SN), a cluster head (CH) and a cluster bridge (PC) figure 4. Within a cluster, the cluster bridge receives the data entering the cluster and is the furthest node from the sink among the cluster nodes. The Cluster Head is the closest node to the sink among the cluster nodes. All cluster formation is done manually by the network administrator. The CH, PC and SN are chosen manually by the administrator. This last property constitutes the whole problemof this mechanism, therefore the use of a cluster self-training algorithm would make such a mechanism much more autonomous.

In [1], the authors propose M2CRL, a cluster construction mechanism for strictly linear wireless sensor networks.

2 Clusters Construction Mechanism for Strictly Linear Wireless Sensor Networks (M2CRL)

The M2CRL algorithmassumes a 2-redundant topology on a homogeneous sensor network where all nodes have the samecapabilities and functionalities. The algorithm is based on a strictly linear topology. The comparison of M2CRL with clusteringalgorithms has highlighted the unsuitability of the latter on linear topologies.

The M2CRL algorithm takes place in two phases: a first phase called Discovery phase and a second phase called Cluster construction phase.

The aim of the discovery phase is to enable the sink to discover its neighbours and to classify them in order of proximity. In this phase, the sink broadcasts an announcement message (Hello) in order to start the discovery process.After receiving the Hello packet, the Neighbours broadcast, in turn, Hello reply messages which will be acknowledged upon receipt, allowing each node to know, exactly, the number of neighbouring nodes v it has.

After this hello exchange process, the nodes neighbouring the sink send an answer to the sink with the value v. The sink then uses the different v values received to determine the proximity of its neighbours. After this phase, the sink transmits a construction packet to its nearest neighbour, this transmission will initiate the cluster construction phase.

In the cluster construction phase, the formation of clusters takes place in a progressive and linear manner. Each node chooses the cluster to which it belongs by means of three quantities:

- A noted constant **Pi** which defines the maximum number of nodes per cluster, Pi is chosen randomly by the sink;
- Quantity **Cid** that identifies a cluster;
- A value **Pn** that determines the position of the node in the cluster to which it belongs.

The construction is carried out by a process of exchanging a packet called a construction packet containing the three quantities Pi, Pn and Cid. These construction packets are acknowledged after receipt.

Initially the sink chooses, randomly, the value of Pi in the interval [3, N/2] N being the cardinality of the network and 3 the degree of redundancy K to which we add 1, in the case of a 2- redundant LWSN we obtain 3. The sink then sets the parameters Cid and Pn to 1 and after defining these parameters, the sink sends to the nearest node a construction packet with the values of the three quantities. As soon as it receives the acknowledgement from this node, the sink sends a new construction packet to the second nearest node with the values Pi, Cid and Pn+1. After having received the construction packets emitted by the sink, the Sink's neighbours set their Pi Pn and Cid to the values contained in the construction packets received from the sink and broadcast, in their turn, a construction packet with the values of Pi, Cid and by incrementing Pn by more than 1.

The rule is that, after having chosen its cluster, each node broadcasts a construction packet to its neighbours with the values Pi, Cid and Pn +1. The choice of the belonging cluster, for a given node, is based on the three principles below:

1. Principle N°1: the Pn position of the node is equal to the maximum of the Pn contained in the received construction packets, P n = max(P n_received) and the Node Cid is equal to the Cid received Cid = Cid_received. This first principle is executedby the node if and only if all received Pn are different from Pi.
2. Principle N°2: This process is executed if one of the received Pn is equal to Pi. This principle says that if the node receivesa Pn equal to Pi, then the node compares the Cids contained in the packets, in doing so, two cases are possible.
    a. First case: The packets contain the same Cid, in this case the node self-elects itself as Cluster Head (CH), sets its Pn to 1and diffuses a construction packet to its neighbours with the values Pi, Cid and Pn+1
    b. Second case: the received packets do not contain the same Cid, in this case, the node, in question, is the first neighbourof the Cluster Head, it sets its Pn to 2, keeps the max of the received Cids, Cid = max(Cid_received) and diffuses aconstruction packet to its neighbours with the values Pi, Cid and Pn+1.
3. Principle N°3: The aim here is to avoid singleton clusters. After broadcasting a construction packet, the node triggers a timerwhich stops when the acknowledgement of the transmitted packet is received. However, if the timer runs out (acknowledgement not received) the node self-elects itself as Cluster Tail (CT). On the other hand, if a node happens to be both CH andCT, then it is a Singleton Cluster, therefore the node positions itself to the parent cluster by decrementing its Cid and setting its Pn to Pi+1.

The authors also give the position Ps of a node with respect to the sink using **equation (1)**

$$\boldsymbol{P_s = P_i(C_{id} - 1) + P_n} \qquad \textbf{(1)}$$

The cluster construction algorithms proposed in the literature, for the most part, were not suitable for linear topologies. On the other hand, the M2CRL algorithm constitutes a mechanism allowing a self construction in clusters specific to linear networks. However, it remains ineffective in the face of linear networks having junction areas.

3 Clusters Construction Mechanism for Junction linear wirelesssensor networks (2CMJ)

In this article, we propose 2CMJ, a cluster construction mechanism for wireless sensor networks with linear topology and

junction areas. The only self-construction mechanism for clusters, found in the literature (to our knowledge) M2CRL [1], becomes ineffective against a linear topology with junctions. Our mechanism is an extension of M2CRL in the face of the constraint caused by junction areas. The algorithm assumes a homogeneous 2-redundant topology. In a junction areas, too, all nodes are neighbours (within transmission range).

The same construction variables used in the M2CRL protocol version are maintained in the 2CMJ version. A new variable Cp (Cluster position) indicating the position of the cluster in relation to the sink has been introduced in this new version.

I. Algorithm:

2CMJ consists of two phases: a first phase called the node definition and neighbourhood discovery phase and a second phase called the cluster formation phase.

A. In the first phase (**Algorithm 1**), each node determines in which areas it is located and consequently to deduce its nature(simple node or junction node). Indeed, in an LWSN with junctions, there are mainly two areas: the strictly linear area inwhich the nodes have, at most, 2*k neighbours, k being the degree of redundancy (in our case four neighbours). There are also junction areas in which the nodes within them have at least 2*k neighbours Therefore, we define two types of nodes: simple nodes (SN), i.e. those belonging to strictly linear areas, and junction node (JN), i.e. nodes located in junction areas. Initially each node broadcasts a Hello message:

a. If a node receives a maximum of four Hello packets from different neighboring nodes, Hello_received ≤ 4, then thenode self-elects as a simple node (SN).

b. If the number of Hello packets received from different neighboring nodes is strictly greater than 4, Hello_received >4, then the node self-elects as a junction node (JN).

After defining its nature the node communicates it to its neighbours and saves all it neighbours in a list.At the junction areas, we define another type of node, in addition to the junction nodes (JN), these are the gate nodes (GN). The gate node is the node connecting a junction area to a strictly linear area, i.e. a packet leaving a junction area will lastly encounter a gate node. If it is an incoming packet, the first node to receive this packet will be the GN. The number of GNs in a junction zone is proportional to the complexity of the zone. The higher the complexity of the zone, the higher the number of NGs in the zone.

Simple nodes (SN) having a (JN) neighbour communicate the list of their neighbours in broadcast, the NJ neighbours receiving such a packet check if, among the transmitting nodes, there are two of them which are one-hop neighbours (close neighbour). If this is the case, then the node self-elects itself as a gate node (GN) (**Algorithm 2**).

Once the (GNs) have been chosen, the second phase can begin.

B. In the second phase, the clusters are gradually constructed in a linear manner. Two types of clusters can be distinguishedhere: Simple Clusters (SC) made up solely of simple nodes, and Junction Clusters (JC) made up solely of junction nodes.This phase of cluster construction is based on two stages:

a. Step 1 reuses the M2CRL algorithm [1] for cluster formation at the level of strictly linear areas **(Algorithm 3) (Algorithm 4)**

b. At the Junction Cluster (JC) level, exactly as in M2CRL, the node closest to the sink will be elected Cluster Head(CH). The latter also coordinates the cluster creation process within the JC in the manner described below **(Algorithm 5)** : Once set, the CH triggers a search for gate nodes by broadcasting a Hello-GN packets. Only the GNs will respondto this packet in unicast to the CH. The CH then registers the different GNs in its cluster.At this stage, the CH knowsperfectly well the nature of its neighbours, the GNs and the NGs. At this stage, the CH knows perfectly well the natureof its neighbours (the GNs and the JNs). Following this, the cluster Head sends two types of construction packages tothe cluster:

1. A classical construction packet, like those used in M2CRL, transmitted in multicast to neighbouring non-GN nodes.The latter, initialise their Pn to 2, the other variables (Cid, Cp) are equal to those contained in the construction packet received from CH.
2. A second construction packet for the different (GN) of the cluster. To each GN, the CH sends a construction packetcontaining an index *i* (unique for each GN) which will be used for the calculation of the Cid of the next Simple Cluster. The (GNs) set their Pn to 2, the other variables (Cid, Cp) are equal to those of the CH. Each GN calculates the Cid (x) of the next Single Cluster (SC) connected to it using the equation **(Equation 2)**. The next SC sets its Cid to the value (x) contained in the construction packet transmitted by the GN **(Algorithm 6)**. Construction continues in strictly linear areas.

$$x = C_{id} + \left(\frac{p}{P_i}\right)^i \quad with \begin{cases} C_{id} & \text{CJ identifier} \\ p & \text{LWSN depth} \\ P_i & \text{Simple Cluster (SC) cardalite defined by the sink} \\ i \geq 1 & \text{index received by the gate node (GN)} \end{cases}$$

II. In order to determine some of the sensor characteristics in the LWSN, we established a mathematical model based on settheory.

Indeed, a linear wireless sensor network can be assimilated to a set of elements arranged in a linear geometry. Consequently, certain mathematical properties that apply to set theory become applicable to LWSN. We have defined a notation system **(Table 1)** that perfectly reflects the correlation between a network of linear wireless sensors network and a set of finite elements. For any node n belonging to a cluster $C_x$of the network, its position $P_{n\to s}$ (distance) from the sink is given by **(Equation 3)**

$$\forall n \in C_x; P_{n\to S} = P_n^x + 2 \times (CJ_{C_x}) + \sum_{CS_i \in S_{C_x}} (C_{CS_i})$$

Table 1: Table of Notation

Notation	Meaning
$N = \{.....\}$	The set of nodes in the WSN
$C = \{.....\}$	The set of clusters in the WSN
P_n^x	The position of node n in cluster x
$F_{C_x}=\{.....\}$	the set of Clusters (son of C_x) belonging to the line leading to the SINK
$J_{C_x} = \{.....\} \subset F_{C_x}$	The set of junction clusters belonging to F_{C_x}
$S_{C_x} = \{.....\} \subset F_{C_x}$	The set of simple clusters belonging to F_{C_x}
$J_{C_i} \cup S_{C_i} = F_{C_i}$	
C_{C_i}	the cardinality of the cluster C_i
C_{CJ_i}	the cardinality of the junction cluster CJ_i
C_{CS_i}	the cardinality of the simple cluster CS_i
CJ_{c_x}	the cardinality of the set J_{C_x}

Algorithm 1: Phase of defining nodes and discovering the neighborhood

```
Result: First phase: Here, each determine its nature and that of its neighbors
Data: neighbor: list_neighbors() int: number_of_neighbors bool: isSN , isJN, isGN
Initialization: number_of_neighbors = 0, isSN=true, isJN=false, isGN=false
struct {
    MACAddress source
    Boolean type
} neighbor
begin
    if (!isSINK) then
        Pcst=Packet(Hello)
        Send(Pcst, Broadcast)
        start(nature_determination_Timer)
        if (Receive_Hello) then
            number_of_neighbors++
    if (End(nature_determination_Timer)) then
        if (number_of_neighbors > 4) then
            isJN= true                                  // the node is located in a junction area
            isSN= false
        Pcst=Packet_Type(isJN)
        Send(Pcst, Broadcast)                  // the packet communicates its type to its neighbors

        if (receive(Packet_Type)) then
            neighbor x
            x.source= Packet(source_Address)
            x.type= Packet(isJN)
            Add(x, list_neighbors)
```

Algorithm 2: gate node definition phase

```
Result: at the end of this phase, the gate nodes are defined
Data: neighbor: List1(), List2() int: a
Initialization: a = 0
struct {
    MACAddress source
    bool type
} neighbor
begin
    if (isSN) then
        for (x ∈ list_neighbors) do
            if (x.type==true) then
                Pcst=Packet_list_neighbors(list_neighbors)
                Send(Pcst, x.source)        // the node send its list of neighbor to its NJ neighbors
    if (receive(Packet_list_neighbors)) then
        if (a==0) then
            List1 ←Packet_list_neighbors(list_neighbors)
        else if (a==1) then
            List2 ←Packet_list_neighbors(list_neighbors)
            start(gate_node_Timer)
        a++
    if end(gate_node_Timer) then
        /* After receiving neighbor lists from neighboring SN                    */

        if (a==3) then
            IsGN = true
            /* the NJs having received only two announcement packets self-elected as gate node
               */
```

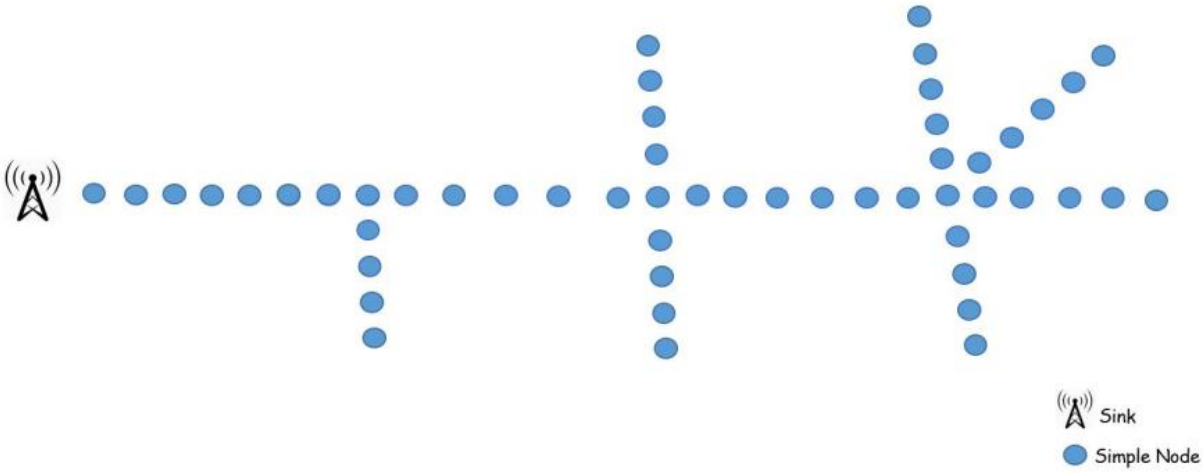


Figure 5: Initial topology before creation of clusters

Algorithm 3: Discovery phase of Sink

Result: First phase: Allow the Sink to discover its neighbors and their degree of proximity
Data: neighbor: Sink_neighbors() int: v

```
Initialization: v=0
struct {
    MACAddress source
    int v
} neighbor
begin
    if (isSINK) then
        Pcst=Packet(Discovery)
        Send(Pcst, Broadcast)
        if (Receive_Response_Discovery) then
            neighbor x
            x.source= Address_Packet
            x.v= v_Packet
            Add(x, Sink_neighbors)
    else
        if (Receive_Packet_Discovery) then
            Pcst=Packet(Hello)
            Send(Pcst, Broadcast)
            start(neighborhood_discovery_Timer)
        if (Receive_Hello) then
            Pcst=Packet(Ack_hello)
            Unicast(Pcst, Sender_Packet)
        while (Receive_Ack_Hello) do
            v++
        if (End of neighborhood_discovery_Timer) then
            Pcst=Packet(Response_Discovery)
            Unicast(Pcst, SINK)
```

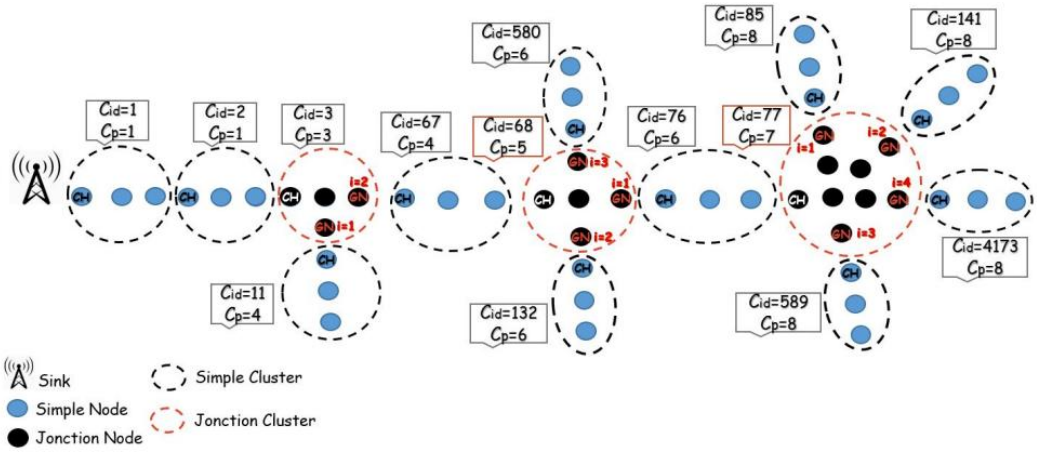


Figure 6: Topology after creation of clusters

Algorithm 4: Cluster construction phase: on strictly linear areas

Result: this phase describes the creation of clusters in strictly linear areas
Data: Int:3< max ≤ N/2 (N is the number of nodes in WNS) ack, Pi, Pn, Cid, Cp
Data: Int : $neighborsID[2], neighborsCid[2]$

```
Initialization: ack=0
begin
    if (isSINK) then
        Pi = random(3, max)
        Cid = 1
        Cp = 1
        Pn = 1
        Pcst=Packet(Pi,Pn,Cid,Cp)
        Send(Pcst, A)          // A is the first closest node to SINK: with the greatest value of v

        if (Receive Ack) then
            Pcst=Packet(Pi,2,Cid,Cp)
            Send(Pcst, B)                                  // B is the second closest node to SINK

    else
        if (Receive_two_Construction_Packet_from_SN) then
            if (isSN) then
                EXECUTE M2CRL       // The M2CRL algorithm is executed in strictly linear lines

            else
                EXECUTE Algorithm 5
        if (Receive_two_Construction_Packet_from_JN) then
            isCH=true
            Pi = Pi(Received_Packet)
            Cid = max(received_Cid)                        // i.e. the x calculate by GN
            Cp = max(received_Cp) + 1
            Pn = 1
            Pcst=Packet(Pi,Pn,Cid,Cp)
            Send(Pcst, Broadcast)
        if (Receive_Ack) then
            ack = 1
            Cancel(Ack_Timer)
```

Table 2: Simulation parameters

Distance between nodes	Number of Nodes	MACProtocol	Radio Module	Power Tx	Sensibility	Simulation Time
500 meters	[52]	TMAC	CC2420	-5dBm	-95 dBm	500s

Algorithm 5: Cluster construction phase: on junction areas

Result: this phase describes the creation of clusters in junction areas
Data: Int:3< max ≤ N/2 (N is the number of nodes in WNS) ack, Pi, Pn, Cid, Cp
Data: $Int : ist\_of\_GN()$

```
Initialization: ack=0
begin
    if (Receive_two_Construction_Packet_from_two_SN) then
        /* the node closest to the SINK is elected CH                    */
        isCH=true
        Pi = Pi(Received_Packet)
        Cid = max(received_Cid) + 1
        Cp = max(received_Cp) + 1
        Pn = 1
        Pcst=Packet(Hello_GN)
        Send(Pcst, Broadcast)                          // To discover the GNs

        start(response_from_GN_Timer)
    if (receive_Hello_GN) then
        if (isGN) then
            Pcst= Packet(Response_Hello_GN)
            Send(Pcst, CH)                  // CH is the sender of the received Hello_GN packet

    while (Response_Hello_GN) do
        Add(source(Pcst), list_of_GN)
    if (End(response_from_Timer)) then
        for (int : i = 1, i <= length(list_of_GN), i++) do
            Pcst= Packet_2(Pi,2,Cid,Cp,i)
            Send(Pcst, list_of_GN(i))
                                  // the packet also contains a unique index i for each GN

        for (int : i = 1, i <= length(list_neighbors), i++) do
            if (list_neighbors(i).type) then
                /* the other nodes of the junction cluster receive construction packets with
                position 2 */
                Pcst= Packet_1(Pi,2,Cid,Cp)
                Send(Pcst, list_neighbors(i).source)

    if (Receive_Packet_1.(destination)==self) then
        /* reception of non-GN nodes from CHs                            */
        Pi = Pi(Received_Packet)
        Cid = Cid(received_Packet)
        Cp = Cp(received_Packet)
        Pn = 2
        Pcst= Packet(Pi,Pn,Cid,Cp)
        Send(Pcst, Broadcast)
```

Algorithm 6: Cluster construction phase on junction areas: Behavior of a junction Node(JN) following a receipt of a Construction Packet(Pcst) from CH

Result: .....

Data: Int: x (x constitutes the cid of the next simple cluster neighboring the GN), Pi, Pn, Cid, Cp, ind

```
begin
    if (Receive_Packet_2.(destination)==self) then
        /* reception of GN nodes from CHs                                   */
        Pi = Pi(Received_Packet)
        Cid = Cid(received_Packet)
        Cp = Cp(received_Packet)
        Pn = 2
        ind = i(received_Packet)          // i is the index of the node, sent by the CH
        x = Cid + (P/Pi)^ind                          // P is the depth of the network
        Pcst= Packet(Pi,Pn,x,Cp)
        Send(Pcst, Broadcast)
```

4 Simulations and results

To test and evaluate 2CMJ, simulations were carried out on the Castalia environment, which is a wireless sensor network simulator based on the OMNet++ platform. The simulation parameters used are listed in **table 2**. Some of these parameters were chosen to allow a 2-redundant network.

We executed the 2CMJ algorithm on a topology of 52 nodes (51 sensor nodes and the sink) with three junction zones (Figure5). The topology used is 2-redundant in the strictly linear zones. The junction zones have complexities of 2, 3 and 4 respectively.

The algorithm is executed with Pi equal to 3. Figure 6 shows the network after clusters creation.

The results in figure 7 give the evolution of cluster formation in relation to the number of nodes in the network. The results show a perfect cluster construction with zero orphan nodes. All the nodes of the network succeed in integrating a cluster, hence the total absence of orphan nodes.

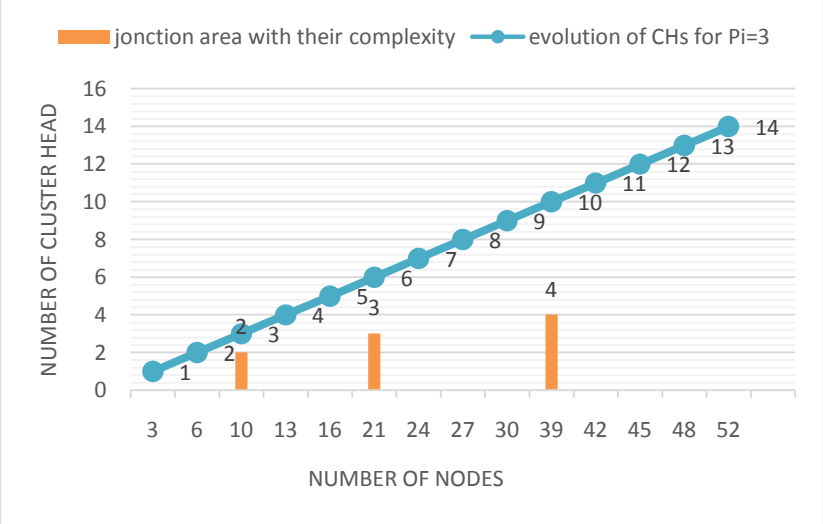


Figure 7: Evolution of CHs created compared to the number of nodes

In the figure 8, we give the evolution of the construction time of the clusters and that of the reference time taken. With 2CMJ, the first cluster has been constructed after a duration of 6.1 seconds after the sink discovery phase, this interval has been taken as reference time, the curve named reference time shows us its evolution, i.e. what would be the construction time of 2CMJ if allthe clusters had exactly a construction time equal to 6.1 seconds. The results obtained show that 2CMJ has a higher evolution than the reference time on the first four clusters formed. From the fifth cluster formed, the evolution remains lower than the reference time. This evolution undergoes a slight increase in the junction zones. This slight increase is caused by the discovery phase of the gate nodes (GN) carried out by the CH in the junction cluster.

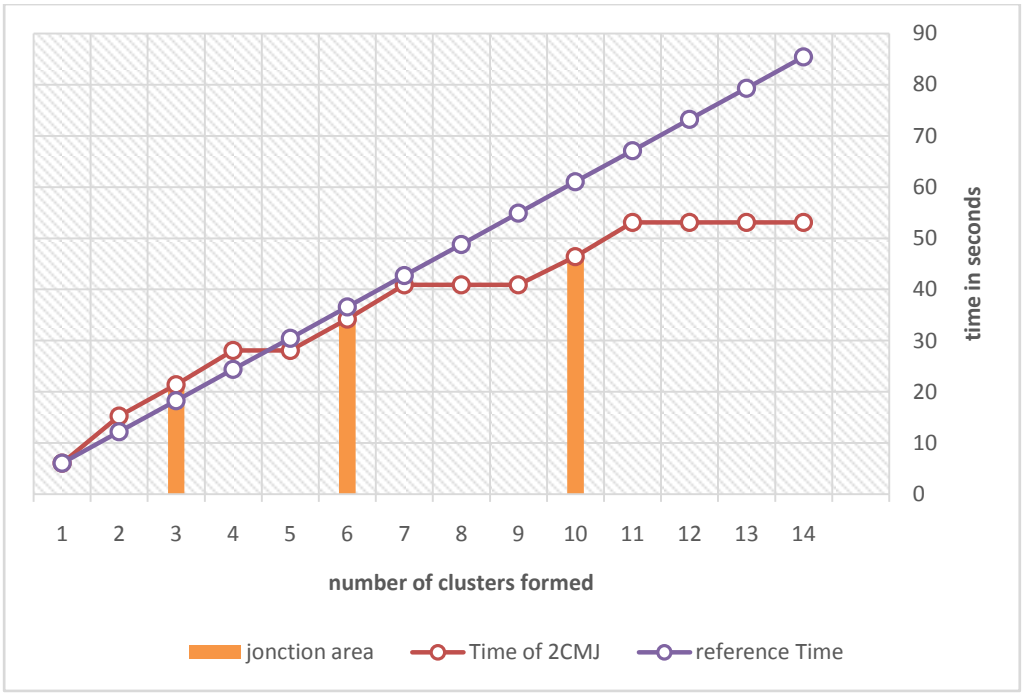


Figure 8: Evolution of the construction time of 2CMJ

Figure 9 shows the position of the clusters represented on a curve called cluster position. With this curve we can see that all clusters with the same position were formed at the same time, even if they are not on the same linear axis. These are the clusters located on positions 4, 6 and 8. We can also notice, graphically, that these clusters are located just after junction areas, their number is defined by the degree of complexity of the junction cluster from which they emanate.

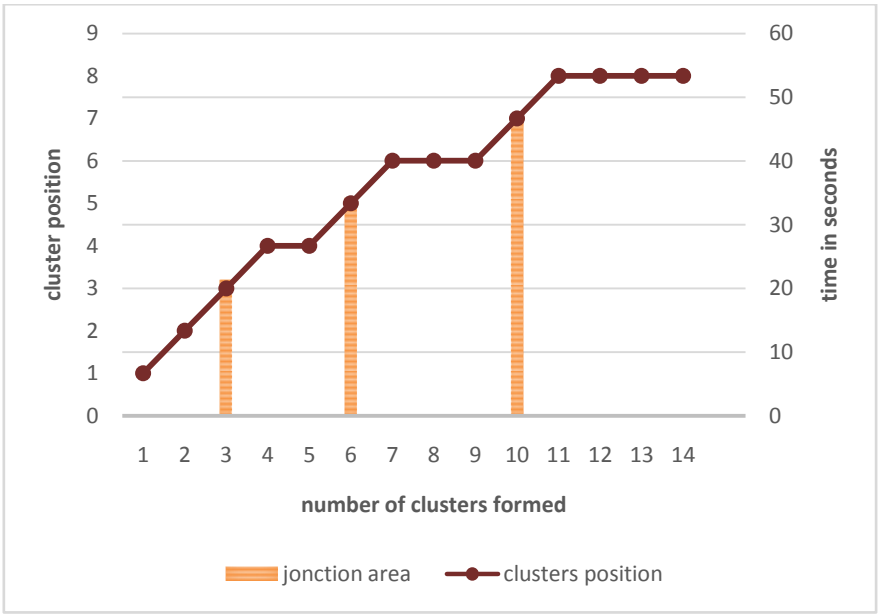


Figure 9: Evolution of the construction time of clusters in relation to their position

5 Conclusion and Perspectives

In this article, we have proposed a self-clustering mechanism for linear wireless sensor arrays with junction zones (2CMJ). The self-constructed cluster algorithms, studied so far, to our knowledge, suffered from an incompatibility with linear topologies. The M2CRL algorithm was able to solve this problem by providing a mechanism exclusively adapted to linear topologies. On the other hand, the major drawback of the latter is the

fact that it does not take into account linear topologies with junction zones, which is moreover very common in linear infrastructures.

The 2CMJ mechanism was proposed in order to resolve the problem at the level of the junction zones. The solution shows excellent results with a success in building clusters to the order of 100% with zero orphan nodes. The algorithm also eliminates the existence of singleton clusters. 2CMJ thus becomes, to our knowledge, the first self-clustering mechanism for linear topology networks with junction zones.

2CMJ can, of course, be improved. It is with this in mind that we plan, in perspective, to make some improvements to the mechanism, such as managing the addition or removal of a node in an existing network. The algorithm can also be improved by bringing a dynamic aspect to the cluster (evolution of a cluster with a change of ID, change of a node type, addition of a branch of nodes). As a result, the clusters will no longer remain fixed but they will vary periodically.

References

[1]. Fall, A., Sarr, M. D., & Sarr, C. (2020, March). Clusters Construction Mechanism for Strictly Linear Wireless Sensor Networks. *In International Conference on Innovations and Interdisciplinary Solutions for Underserved Areas* (pp. 224-237). Springer, Cham.

[2]. S. Varshney, C. Kumar, A. Swaroop, (December 2017) Leach Based Hierarchical Routing Protocol for Monitoring of Overground Pipelines Using Linear Wireless Sensor Networks. J *In: 6th International Conference on Smart Computing and Communications*, ICSCC 2017, 7-8 December 2017, Kurukshetra, India

[3]. Y.-C. L. Meng-Shiuan Pan, Hua-Wei Fang and Y.C. Tseng, (2008). Address assignment and routing schemes for zig bee-based long-thin wireless sensor networks, *In IEEE Vehicular Technology Conference 08*, Spring 2008, 2008, pp. 173177.

[4]. Stoianov, L. Nachman, S. Madden, and T. Tokmouline, (2007) Pipenet: A Wireless Sensor Network for Pipeline Monitoring, *In ACM IPSN*.

[5]. Marco Zimmerling, Waltenegus DARGIE et Johnathan M. REASON. (2008) Localized power-aware routing in linear wireless sensor networks. *In: Proceedings of the 2nd ACM international conference on Context-awareness for self-managing systems. CASEMANS 08 New York, NY, USA: ACM,*2433 (cf. p. 1, 3, 35)

[6]. Abbasi A, Younis M (2007). A survey on clustering algorithms for wireless sensor networks. *Comput Commun 30:28262841*

[7]. Konrad Lorincz, Matt Welsh, Omar Marcillo et al. (2006) Deploying a wireless sensor network on an active volcano *In: IEEE Internet Computing 2006*, p. 1825 (cf. p. 1, 11)

[8]. E. H. M. Ndoye, F. Jacquet, M. Misson et I. Niang. (2014) Using a token approach for the MAC layer of linear sensor networks: Impact of the node position on the packet delivery. *In: 2014 IFIP Wireless Days (WD). 2014*, p. 14 (cf. p. 38, 70).

[9]. Imad Jawhar, Nader Mohamed and Dharma P. Agrawal. (2011) Linear wireless sensor networks: Classification and applications. *In: Journal of Network and Computer Applications 34.5 (2011)* p. 16711682 (cf. p. 3, 5, 35, 38, 39, 59, 70).

[10]. Ali Shokouhirostami, HengamehKeshavarz, FarahnazMohanna, Ali AsgharRahmaniHosseinabadi, (2017) Survey on Clustering in Heterogeneous and Homogeneous Wireless Sensor Networks .*The Journal of Super computingů August 2017* See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/319155887.

[11]. Dali Wei, H Anthony Chan, Shaun Kaplan (june 2008) Energy efficient Clustering algorithms for Wireless Sensor Network. *Conference Paper. DOI:10.1109/ICCW.2008.50 Source: IEEE Xplore*.

[12]. Sukun KIM, Shamim PAKZAD, David CULLER et al. (2007) Health monitoring of civil infrastructures using wireless sensor networks *In: Proceedings of the 6th international conference on Information processing in sensor networks. ACM Press,*2007, p. 254263 (cf. p. 1, 3, 12)

[13]. R. Baker, K. Armijo, S. Belka et al. Wireless sensor networks for home health care (2007). *In: Advanced Information Networking and Applications Workshops, 2007 AINAW07. 21st International Conference on. T. 2. 2007,* 832837 (cf. p. 1, 1)

[14]. Heinzelman, W.R.; Chandrakasan, A.; Balakrishnan, H., (2000) "Energy efficient communication protocol for wireless microsensor networks," *In: System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference* vol., no., pp. 10 pp. vol. 2,, 4-7 Jan. 2000