# Determination of Itinerary Planning for Multiple Agents in Wireless Sensor Networks

Rais Amine[1], Bouragba Khalid[1] and Ouzzif Mohamed[1]

[1] Laboratory RITM, ENSEM, Hassan II University Of Casablanca, Casablanca, Morocco

***Abstract:*** The mobile agent is a new technology in wireless sensor networks that outperforms the traditional client/server architecture in terms of energy consumption, end to end delay and packet delivery ratio. Single mobile agent will not be efficient for large scale networks. Therefore, the use of multiple mobile agents will be an excellent solution to resolve the problem of the task duration especially for this kind of networks. The itinerary planning of mobile agents represents the main challenge to achieve the trade-off between energy consumption and end to end delay. In this article we present a new algorithm named Optimal Multi-Agents Itinerary Planning (OMIP). The source nodes are grouped into clusters and the sink sends a mobile agent to the cluster head of every cluster; which migrates between source nodes, collects and aggregates data before returning to the sink. The results of the simulations testify the efficiency of our algorithm against the existing algorithms of multi-agent itinerary planning. The performance gain is evident in terms of energy consumption, accumulated hop count and end to end delay of the tasks in the network.

***Keywords*:** Mobile agents, wireless sensor networks, energy efficiency, end to end delay, itinerary planning.

## 1. Introduction

The client/server architecture scheme in the wireless sensors networks (WSN) isn't quite efficient in terms of energy consumption; each sensor node has to relay its collected data through its neighbor sensor all the way up to the sink. The whole network becomes then vulnerable due to the fast energy depletion at the sensor level.

Another downside linked to the client/server architecture is the latency issue: the delay between the data emission time and reception time which can be intolerable, especially in the large scale networks. This lag results, on one hand into inaccurate sink feedback actions based on expired sensed data; on the other hand into unsuitable sink action commands to the targeted nodes, making them irrelevant to the actual sensed field situation.

To avoid all these problems we propose to use mobile agents for efficient energy saving all over the network. The mobile agents (MAs) gather and process data and finally transmit them to the sink. The process operation can be performed locally and only processed data can be sent to the sink, thus the energy consumption can be reduced.

The deployment of single MA in a small network will resolve the issues of the limited bandwidth in WSN and the reduced lifetime of the batteries of the sensors. However, for a large scale network the tasks duration will exceed the acceptable delay time, for that we can use multiple MAs that gather in parallel the data of the network. Every MA derives an optimal itinerary among the source nodes; the route taken during agent migration can have a significant impact on the power consumption. Thus, the multi-agent itinerary planning (MIP) is the key to improve the performance gain in the MAs architecture.

In WSN the neighboring nodes sense almost the same data. Therefore, it's a waste of time and energy to move the MA between all the nodes of the network to gather data. The big number of nodes to cross will increase the task duration, this latter will take an unacceptable value, thus the MA must migrate to the farther nodes in its transmission range to avoid the redundancy and get a higher degree of information gain [1]. The existing algorithms of MIP propose to arbitrary distribute the source nodes in the network, but in our algorithm we divide the network into clusters, we look for the source nodes and the intermediate nodes from the nodes of every cluster, and the optimal itinerary of the MA between them.

The rest of this paper is arranged as follows: Section 2 discusses the related works concerning mobile agents and MIP problem. In Section 3, we present the client/server architecture and the mobile agents in WSN. The Optimal Multi-Agents Itinerary Planning algorithm is explained in Section 4. We show the simulation results in Section 5 and finally we conclude this paper in Section 6.

## 2. Related works

The authors of the article [2] use a Directional Source Grouping Algorithm for Multi-agent Itinerary Planning (DSG-MIP) to divide the network into directional sector zones with an angle threshold θ; the value of this angle is important to achieve the balance between energy cost and task duration. The source nodes in the same sector are grouped to form an itinerary planning for the MA; the number of MAs in the network is equal to the sum of source nodes in the range transmission of the sink. These nodes represent the starting points for the MAs in their travelling among nodes to gather data in the sectors.

Damianos Gavalas et al. propose another MIP algorithm in [3, 4], using an Iterated Local Search (ILS) to find the attached itinerary of every node in the network. Like DSG-MIP, the nodes in the range of the sink are the starting nodes of the itineraries; the remaining nodes in the previous MIP algorithms consider attaching only at the end of the itineraries. In contrast, ILS tries to attach these nodes at any possible position in the objective to minimize the energy cost and create efficient itineraries of MAs to extend the lifetime of the network.

Another approach of routing that looks to the mobile agent itinerary is proposed by Husna et. al in [5], The Ant-based routing algorithm use the pheromone update technique, it is an improvement of Ant Colony System (ACS), its main objective is to find an optimal itinerary for routing in WSN and discover the alternative path for the transmission of packets to their destination node.

The authors of the article [6] propose the use of MAs and show the limits of client/server architecture. To find the optimal itinerary of single MA they propose two algorithms

for single itinerary planning: Global Closest First (GCF) and Local Closest First (LCF). In the GCF algorithm the MA tries to reach the next closest sensor node to the center of the network, while in the LCF it looks for the nearest node to the current sensor node.

The itinerary of MA is important to reduce the power consumption. Min Chen et al. describe in the article [7] the problem of itinerary planning of multiple MAs in WSN. For each MA they present the IEMF (Itinerary Energy Minimum for First-source-selection) which is an extension of LCF. This algorithm focuses on the choice of the first source node to be visited by the MA; it gives positive results in term of energy consumption. An iterative version of this algorithm: IEMA (Itinerary Energy Minimum Algorithm) will be used to optimize the remaining source nodes for the entire itinerary in the network. The authors also present a generic framework for MIP algorithm in WSN, which includes: determination of the number of MAs, the subset of source nodes affected to every agent and the itinerary planning of each MA.

The authors in [8] present the disadvantage of single agent based itinerary planning (SIP) in large scale networks. They propose to use multi-agent based itinerary planning: CL-MIP (Center Location-based Multi agents Itinerary Planning) algorithm to resolve the problem of delivery delay task. In MIP the visiting area of MA is a circle/oval centered at the visiting central location (VCL), which is the center of this area with high source node density. All the source nodes within the circular area form a group will be assigned to the visiting list of the corresponding MA. If there are uncovered source nodes the same process will be repeated to cover all source nodes in the network. The path of a MA inside a group will be determined by one of the existing SIP algorithms like: GCF, LCF, IEMF and IEMA.

A spanning tree algorithm named Minimum Spanning Tree for Multi-agent Itinerary Planning (MST-MIP) is used in [9] to create groups of nodes and find the optimal itinerary planning of multiple agents. All the source nodes of the network are modeled as totally connected graph (TCG); the vertices are represented by the source nodes and the weight of each edge is equal to the number of hop count between two nodes. The source nodes in the branch of the tree are grouped and the MA is dispatched to move among them before returning to the sink with the gathered data. The number of MAs in the network is determined by the number of direct vertices connected to the sink. The authors introduce a balanced version of MST-MIP named balanced spanning tree for multi-agent itinerary planning (BST-MIP), they use a balancing factor α to achieve the tradeoff between energy consumption and delivery latency of the MA. Furthermore, there are some other tree based algorithms for MIP problem like: NOID [10], CBID [11], and TBID [12].

A Genetic Algorithm for Multi-agent Itinerary Planning (GA-MIP) is studied in [13]; the authors encode the number of MA circulated in the network and the subset of source nodes covered by each agent. GA-MIP algorithm creates a performance aware fitness function to get near optimal results of the itinerary planning of the agents. This algorithm is more efficient than the above algorithms of MIP in term of performance metrics but the implementation of GA-MIP is difficult due to the calculation complexity of the itinerary planning.

The most existing MIP algorithms use the location information to determine the itinerary of the mobile agents in the clusters of the network, but they neglect the amount of data provided by each sensor, which will cause an unbalanced data gathered by the MAs. To resolve this problem Imene et.al [14] propose a MIP algorithm named GIGM-MIP (the Greatest Information in the Greatest Memory-MIP), based on the balance between the data size and the location information to optimize the energy consumption and the data collection duration.

## 3. Client/server architecture and mobile agents in WSN

### 3.1 Client/server architecture

The nodes sense data and forward them to the nodes in their range transmission, the sensed data will be transmitted from one node to other until they reach the sink with a multi-hop manner, the neighboring nodes of the sink are critical nodes they are the bridge to the sink, the dead of this node will stop the transmission of data to the sink and collapse the network, then an efficient routing data must be followed to reduce the energy consumption in the network.

The energy consumed to send a message of m bits to a node from d distance is estimated by Heinzelman et.al in [15] (Figure1), it's equal to:

$$E_{TX}(m,d)=E_{elec}*m + E_{amp}*m \quad (1)$$

To receive the same message by a node, the energy consumed would be calculated by this equation:

$$E_{RX}(\text{m})=E_{elec}*\text{m} \quad (2)$$

We denote $E_{elec}$ like the energy dissipated to run the receiver or transmitter circuitry, the transmission amplifier consumes the energy ($E_{amp}$), it would be equal to:

$$\begin{cases} Eamp = \theta_{fs} * d^2 & \text{if d} \le l_0 \\ Eamp = \theta_{mp} * d^4 & \text{if d} > l_0 \end{cases} \quad (3)$$

Where $l_0=\sqrt{\frac{\theta_{fs}}{\theta_{mp}}}$, $\theta_{fs}$and $\theta_{mp}$ are the radio amplified consumed energy for free space propagation and multipath fading channel model. According to Heinzelmen et al. in [16] the value of $\theta_{fs}$and $\theta_{mp}$ are respectively 10pJ/bit/$m^2$ and 0.0013pJ/bit/$m^4$.

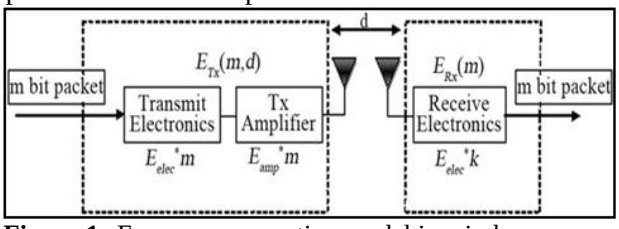


**Figure 1.** Energy consumption model in wireless sensor networks[15]

The communication between sensor nodes consumes the biggest quantity of energy; it will deplete the remaining energy of sensors. In the large scale networks the client/server architecture isn't efficient, the management of energy consumption will become difficult with presence of thousands of messages that circulates in the network, which causes collisions, exceeds the bandwidth of the network and reduce the lifetime of the network, the MAs will be an excellent solution to resolve these problems.

### 3.2 Mobile agents in wireless sensor networks

To gather data in the MA architecture the sink dispatches in the network the MA, this latter is a code program that circulates between sensor nodes, it gathers, processes and aggregates data locally before returning to the sink. To integrate MA in WSN the authors of [17] use the platform of Agilla middleware for TinyOS [18] sensors to support self adaptive application. Beside Agilla, there are another middlewares named ActorNet [19,20] and two java-based platforms to program MA in WSN for Sun SPOT[21] sensors, Mobile Agent Platform for Sun SPOT Sensors (MAPS)[22] and Agent Factory Micro Edition (AFME) [23,24].

The source nodes sense data and have to send them to the sink; the MA crosses all these nodes to gather data, while the intermediate nodes between source nodes can only forward the MA packet (Figure 2).

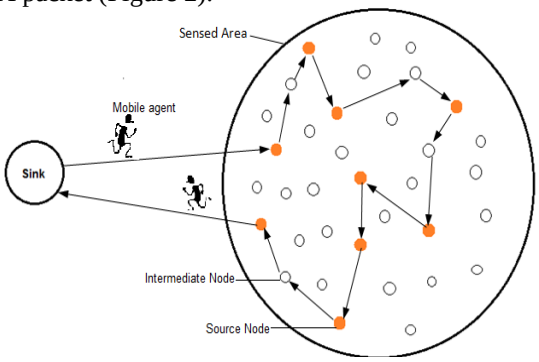


**Figure 2.** Architecture of mobile agents in WSN

The bandwidth of WSN is lower than the wired network and data traffic may exceed the network capacity. The MA will be an excellent solution for this issue, since it can reduce the huge number of sensory data by eliminating the redundancy and avoid communication overhead.

The energy needed for transmitting and receiving a MA during migration between two source nodes m and n is equal to:

$$E_t^{m,n} = (e_t * S_m) + O_t^{clt} \quad (4)$$

$$E_r^{m,n} = (e_r * S_n) + O_r^{clt} \quad (5)$$

Let's $e_t$ and $e_r$ are the spending energy per bit for sending and receiving a MA. $O_t^{clt}$ and $O_r^{clt}$ are the consumed power to exchange control messages, and $S_m$ and $S_n$ are the size of the MA packet at the source nodes m and n.
We consider $D_h$ the size of the raw data in the $h^{th}$ source node, after a local processing by the MA with a reduction ratio Ω ($0 \leq \Omega \leq 1$), the new size of sensed data will be equal to $R_h$. We will express that below:

$$R_h = (1-\Omega) * D_h \quad (6)$$

The aggregation operation will begin from the second source node, the MA compares the data collected from the current node and the previous crossed nodes and eliminates the redundancy .The size of MA packet increases from one source node to another, and it is possible to calculate the size of the MA at a source node j with this formula:

$$\begin{cases} S_1 = S_0 + R_1 \\ S_j = S_1 + \sum_{h=2}^{j} (1-\rho) * R_h \end{cases} \quad (7)$$

With $S_0$ is the MA size when it's dispatched by the sink, it's equal to the size of the processing code of the MA, $S_j$ is the reduced data for collected data by the source nodes and ρ is the aggregation ratio to measure the compression performance with $0 \leq \rho \leq 1$, when ρ is equal to zero, it means that there is no data processing executed by the MA. This operation consumes significantly less in terms of energy than the data transmission [25], thus the energy consumed to process and aggregate data by the MA is neglected.

## 4. The Optimal Multi-agents Itinerary Planning algorithm

The MIP is the main challenge to reduce the energy cost and the task duration of the gathering data in the network, a survey and comparison of the existing MIP in WSN is made in 2010 by X. Wang et.al in [26]. We propose in this section a new algorithm of MIP named Optimal Multi-Agents Itinerary Planning (OMIP); we divide the network into clusters with ECRP algorithm [27] and define the medoid nodes, the sink sends the MAs to the clusters to gathers data from the source nodes; in every cluster we select the list of source nodes to sense data and the lists of intermediate nodes to forward MA between source nodes.

We assume the following properties about our sensor network:

- The sink knows the location of the sensor nodes.
- The computation of the clusters and the medoid nodes is executed at the sink, which owns high resources of computation and energy.
- All sensor nodes are equal in resources (sensing, computation, processing and initial energy level…).
- All the nodes have the same maximum transmission range R.
- The nodes and the sink are stationary: static network.
- The nodes sense data with a periodic manner.

The use of GPS technology to get the location of every node will consume a big amount of energy; it's possible to use trilateration and triangulation techniques [28, 29, 30] to determine the coordinates of every sensor node in the network. For the estimation of the distance between nodes, we will use the Received Signal Strength Indication (RSSI) [31].

### 4.1 ECRP algorithm

ECRP (Efficient Clustering Routing Protocol) [27] is a modified version of clarans algorithm [32, 33] used in data mining for clustering of data; we use this algorithm and adapt it to the WSN environment. The main objectives of ECRP algorithm are to divide the network of n nodes into $\boldsymbol{v}$ clusters ($\boldsymbol{v} \leq$n), select the medoid node ($\boldsymbol{MNi}$) of each cluster Ci, and minimize the average distance between this special node and the other nodes of the cluster ($\boldsymbol{X_j}$). We express this operation by this equation:

$$Min\ avgs \sum_{i=1}^{v} \sum_{j \epsilon Ci} \boldsymbol{d}(\boldsymbol{X_j}, \boldsymbol{MNi}) \quad (8)$$

Heinzelmen et al. define in [16] the value of $v$ with the following equation:

$$v = \frac{\sqrt{n}}{2\Pi} \times \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}} \times \frac{L}{d_s^2} \quad (9)$$

Where $L$ is the side of the sensed square field and $d_s$ is the average distance between sensor nodes of the network and the sink.

In this article, the medoid node is considered as the cluster-head, it's the most centralized node in the cluster and the closest node to the other nodes. For that we choose it like the starting and the arrival node in the tour of the MA between the source nodes of the cluster.

### 4.2 Source nodes selection procedure

The Big number of nodes crossed by MAs will reduce the performance of the MA architecture, specially in large scale networks with hig density of nodes, thus the MAs must get a higher degree of information gain, during their travelling between the source nodes of the network. Thus these nodes must be located and distributed with an ideal manner in every cluster, in our algorithm we try to minimize the distance between the source nodes and the other nodes of the cluster, so that these source nodes are located in the following coordinates:

$$\begin{cases} Si_x = r \times cos(\frac{2\Pi}{N_c} \times (i - 1)) + M_x \\ Si_y = r \times sin(\frac{2\Pi}{N_c} \times (i - 1)) + M_y \end{cases} \quad (10)$$

With i=1, 2, 3,…,$N_C$, and r is the average distance between the medoid node and the other nodes in the cluster, $(M_x, M_y)$ is the coordinate of the medoid node, and $N_C$ is the number of the source nodes in every cluster, it's value will be expressed by:

$$N_C = \frac{N_s}{v} \quad (11)$$

$N_s$ is the total number of the source nodes that sense data in our network and $v$ is the number of the clusters in the network. The source node $S_i$ is the closest sensor node to the coordinate $(Si_x, Si_y)$, thus we can specify the source nodes in every cluster; the list of source nodes of every cluster is calculated centrally by the sink. An example of this operation with $N_C$=4 is illustrated in the Figure 3.

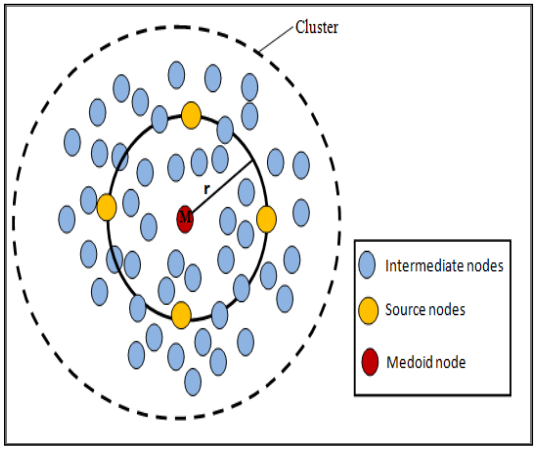


**Figure 3.** Example of source node selection procedure

### 4.3 Architecture of our network

The ECRP algorithm subdivides the network into $v$ clusters. It's possible to determine the value of $v$ with the equation 9; we choose the medoid node of every cluster as a cluster-head. The sink sends a MA to every cluster of the network to gather, reduce and aggregate data of the list of source nodes; it begins its tour from the medoid node and moves among nodes before returning to the same medoid node. Finally, this latter sends the MA with the gathered data to the sink (Figure 4).

We subdivide the itinerary of MA to collect data in every cluster into three stages:

a) Beginning stage: In this stage the sink sends the MA to the medoid node; the MA doesn't gather any data in this phase, its size remains unchanged, and is equal to the size of the processing code, then the energy consumption in this phase($E_{Beg}$) is the energy consumed from the sink to the medoid node of the cluster ($E^{S,M}$):

$$E_{Beg} = E^{S,M} \quad (12)$$

b) Touring stage: it starts from the time the medoid node sends the MA to the first source node; during this phase the MA moves among the source nodes, it gathers, reduces and aggregates data; finally it reaches the last source node and return to the medoid node. Thus, the total of energy cost is the sum of the energy consumed from the medoid node to the first source node ($E^{M,S1}$), the total of energy consumed between the source nodes, and the energy depleted to leave the last source node and return to the medoid node ($E^{N_c,M}$) :

$$E_{Tour} = E^{M,S1} + \sum_{k=2}^{N_c} E^{k-1,k} + E^{N_c,M} \quad (13)$$

c) Returning stage: the MA reaches the medoid node with the gathering data of all source nodes in the cluster, the energy consumed during this stage is the energy cost to send the MA from the medoid node to the sink($E^{M,S}$), then the energy consumed to return to the sink is :

$$E_{Rg} = E^{M,S} \quad (14)$$

Therefore, the total energy consumed by the MA to gather data in a cluster ($E_{itinerary}$) is the sum of the energy depleted during the three stages:

$$E_{itinerary}=E_{Beg}+E_{Tour}+E_{Rg} \quad (15)$$

The itinerary of MA will be specified dynamically with a modified version of LCF algorithm (MLCF) to adapt it to the architecture of our network. The packet structure of MA is defined at Figure 5.

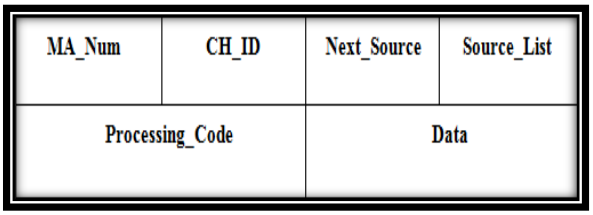


**Figure 5.** Mobile agent packet structure

**Table 1.** Notation of mobile agent packet structure elements

| Symbol | Definition |
|---|---|
| MA_Num | The value of MA_Num is incremented whenever the sink dispatches a new MA |
| CH_ID | The identification of the cluster |
| Next_Source | The next source node to be visited |
| Source_List | The list of source nodes to visit |
| Processing_code | The code to execute in each source node to process the gathering data |
| Data | The gathered information by the sensor nodes |

### 4.4 The itinerary of MAs in OMIP algorithm

To choose the next source node and get the optimal itinerary of the MA inside a cluster we use the LCF algorithm with some modifications. The LCF chooses the closest node to the center of a sub area to be the first node to begin the trip of the MA which returns finally to the processing element (PE) at the end of the trip. In our algorithm the medoid node represents both the center of the sub area (cluster) and the cluster head. The sink sends the MA to the medoid node of the cluster. The MA then moves among the source nodes with the use of a modified version of LCF algorithm (MLCF) and returns to the medoid node with the sensed data.

**Algorithm 1** Modified version of the LCF algorithm (MLCF):

**Notation:**
**$N_c$:** Number of source nodes in the cluster
**k:** index of the current source node, with k∈[0, $N_c$]
**Input:**
**MN**: the medoid node of the cluster
**Source_list**: list of source nodes to visit in the cluster
$SN_k$: Current source node
**Result:** the next source node (Next_Source)
**Begin**
1: $N_c$←sizeof(Source_list);
2: k←indexof($SN_k$);
3: **Switch** the value of k **do**
4: **Case** k=0 /* means the MA is at the medoid node*/
5: Find source node S from Source_list with the smallest distance d(S,MN);
6: Next_Source←S;
7: **Case** k= $N_c$
8: Next_Source←MN;
9: **Otherwise**
10: Find source node S from the rest of Source_list with the smallest distance d(S, $SN_k$);
11: Next_Source←S;
12: **End Switch**
13: **return** Next_Source;
**End**

There will be some intermediate nodes between two source nodes m and n where the distance between m and n ($d_{mn}$) is greater than the maximum transmission range(R) of these nodes ($d_{mn}$>R). We can estimate the minimum number of intermediate nodes $Min(N_{inter})$ between two source nodes m and n by:

$$Min(N_{inter})=\frac{d_{mn}}{R} \quad (16)$$

The intermediate nodes will just forward the MAs between the source nodes without generating any information. When the MA crosses these nodes the size of its data doesn't change and it remains the same. Therefore, the energy consumption of all intermediate nodes between the source nodes m and n is equal to:

$$N_{inter} * (E_t^{m,n} + E_r^{m,n}) \quad (17)$$

The total amount of energy consumed to send a MA from m to n is the sum of the transmitting energy of the node m, the receiving energy of n and the energy consumed by the intermediate nodes between m and n:

$$E^{m,n} = (1+N_{inter}) * (E_t^{m,n} + E_r^{m,n}) \quad (18)$$

To minimize the depletion of energy in the MA architecture for the communication between two source nodes, we have to reduce the number of intermediate nodes and optimize the path of the MA between these nodes. Therefore, the MA leaves the source m to the list of the intermediate nodes (L), and moves between them until it reaches the destination source node n (Figure 6).

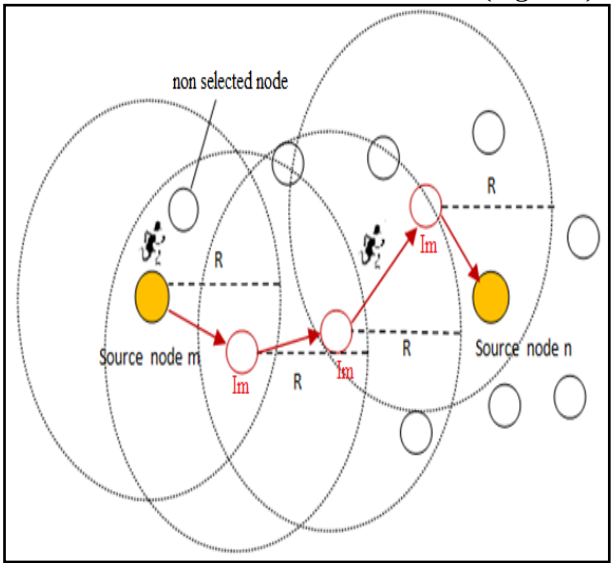


**Figure 6.** Optimized itinerary between two source nodes

The objective of the algorithm2 is to get the list of intermediate nodes (L) crossed by the MA to reach the destination source node. If the distance between the source and the destination node is less than the maximum transmission range R, the list of intermediate nodes will be null, and the MA doesn't have to cross any intermediate node: it will move directly to the next source node.

The next intermediate node should be selected from the set of nodes in the transmission range of the current intermediate node, it is the closest one to the destination source node. The MA moves to this node and look for the next one, thus this process continues until the MA reaches the destination source node.

**Algorithm 2** Itinerary planning algorithm of mobile agent between two source nodes

**Notation:**
**R:** maximum transmission range
**Im:** the current intermediate node
**In:** the next intermediate node
**L**: the list of intermediate nodes to forward data by the MA
$\boldsymbol{W_{Im}}$: the set of nodes in the transmission range of Im
**min:** the minimum distance between the nodes in the transmission range of the current intermediate node, and the destination source node
**Input:**
**m** and **n** are two source nodes with m is the source and n is the destination
**Initialization:**
$W_{Im} \leftarrow \{\}$;
L ←{};
Im←m;
**Result:** the list of intermediate nodes (L)
**Begin**

```
While d(Im,n)>R do /*when the node n is in the transmission range of the intermediate node Im, we will stop */
  W_Im ←{nd with d(Im,nd)<R};/*all the nodes nd in W_Im must be in the transmission range of Im */
  min←large value;
  For each node nd in W_Im do
    Calculate d(nd,n);
    if d(nd,n)<min then /*the node nd is the closest node to the destination node n from all the nodes in the transmission range of Im*/
      In←nd; /*the node nd will be the next intermediate node*/
      min←d(In,n);
    End if
  End for
  Im←In; /* the MA will move to the next intermediate node
  L←L+In; /*we add In to the list L */
End while
Return L;
End
```

The flowchart of the communication process for every MA is presented in the Figure 7, the sink sends the MA to the medoid node of the cluster with the list of source nodes to visit, the choice of the itinerary between these nodes is made with the algorithm1, if the distance between two source nodes is greater than the maximum transmission range (R) the MA has to roam the list of intermediate nodes, else it will move directly to the next source node, then we store the processing data of the source node with new content in the data of the MA. The size and the energy consumption of this latter increase from one source node to another, due to the rising of the size of its data. In the end of the trip the MA returns to the medoid node, and thereafter to the sink with the gathering data of the cluster.

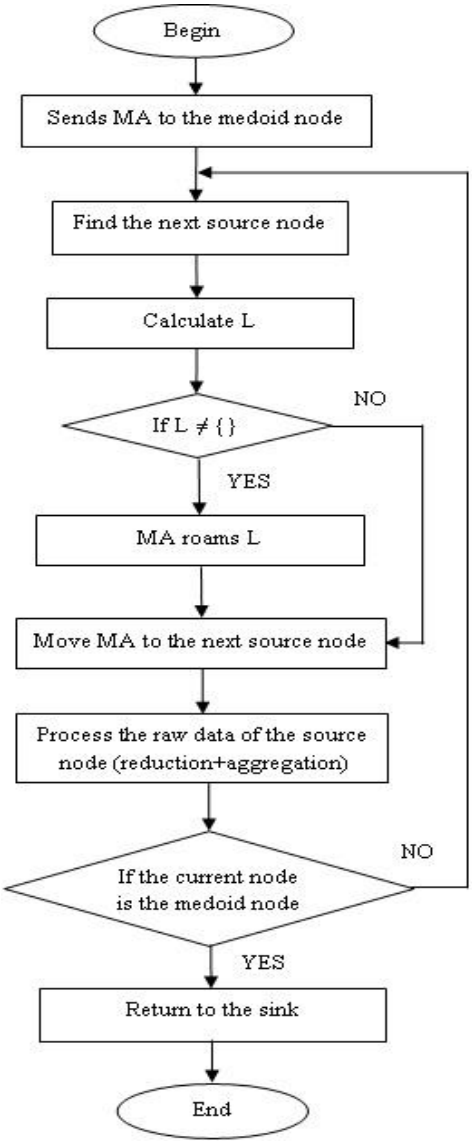


**Figure 7.** Flowchart of the communication process of mobile agents

Algorithm 3 uses algorithms 1 and 2 in order to create the itinerary of the MA in every cluster; it gets the sequence of the source nodes to visit (MLCF function) and the intermediate nodes between these source nodes (intermediate function). The MA elements get changed along the itinerary, the size of the MA data and the energy consumption are calculated at the three stages. In the end, the algorithm 3 returns the itinerary planning followed by the MA to gather data in the cluster.

**Algorithm 3** Itinerary planning of MA in the cluster

**Notation:**
$\boldsymbol{MA_i}$:the mobile agent gathering data of the cluster i
$\boldsymbol{N_c}$: the number of the source nodes in the cluster i

**k:** index of the current source node
$\boldsymbol{SN_k}$: Current source node, with k=0 or k=$N_c$+1 means MA is at the medoid node
**CurrentNode**:the current source node selected by the MA
$\boldsymbol{IP_i}$:itinerary of the MA in the cluster i
**MA_SeqNum, CH_ID, Next_Source, data**; /*see the definition of these variables in the table1*/
$\boldsymbol{S_k, R_k,}$ **ρ,** $\boldsymbol{\Omega, D_k,}$ **L** /*see the definition of these variables above*/

**Input:**
$\boldsymbol{MN_i}$:medoid node of the cluster i
$\mathbf{List\_source_i}$**:**list of source nodes in the cluster i

**Initialization:**
data←{}; /*the value of data is null in the sink*/
k←0;
$N_c$←sizeof(List_source$_i$);
$S_0$←Sizeof(Processing_code); /* The size of the MA at the sink and the medoid node is equal to the size of the Processing_code */
$E_{Tour}$ ←0;

**Result:** the itinerary of the MA in the cluster i

**Begin**
1: Send $MA_i$ to $MN_i$; /*the sink sends the MA to the medoid node of the cluster i*/
2: $E_{Beg} \leftarrow E^{S,M}$;
3: MA_SeqNum←MA_SeqNum+1;
4: CH_ID←i;
5: $SN_0 \leftarrow MN_i$;
6: $IP_i \leftarrow SN_0$;
7: **While** k<$N_c$+1 **do**
8: Next_Source←MLCF ($MN_i$, List_source$_i$,$SN_k$) ;
9: k←k+1;
10: $SN_k$←Next_Source;
11: L←intermediate($SN_{k-1}$, $SN_k$) ;
12: $IP_i \leftarrow IP_i \cup L \cup \{SN_k\}$; /*the list of intermediate nodes and the next source node will be attached to the itinerary*/
13: **If** L is not empty **then**
14: $MA_i$ Roams L;/*the MA have to roam all the intermediate nodes in L before reaching the next source node*/
15: **End if**
16: Move $MA_i$ to $SN_k$;
17: $E_{Tour} \leftarrow E_{Tour}+E^{k-1,k}$;
18: $R_k \leftarrow (1-\Omega) * D_k$; /* we reduce locally the raw data at the source node k */
19: **If** $MA_i$ is empty **then**
20: $S_1 \leftarrow S_0+R_1$; /*we store the content of the first node in the data of $MA_i$ */
21: **Else**
22: **If** $SN_k$ has new data **then** /*this operation begins from the second source node*/
23: $S_k \leftarrow S_1+\sum_{h=2}^{k}(1-\rho) * R_h$; /*we store the sensed data after aggregation in the MA*/
24: **Else** /*the current source node hasn't sensed any new data*/
25: $R_k$ ←Null; /*the redundant data will not be stored*/
26: $S_k \leftarrow S_{k-1}$; /*the stored data remains the same as in the previous source node*/
27: **End if**
28: $E_{Tour} \leftarrow E_{Tour}+E^{k-1,k}$;
29: **End if**
30: **End While**
31: Return $MA_i$ to the sink; /*in the end of the trip the MA returns to the medoid node */
32: $E_{Rg} \leftarrow E^{M,S}$;
33: $E_{itinerary} \leftarrow E_{Beg}+E_{Tour}+E_{Rg}$;
34: **Return** $IP_i$;
**End**

The algorithm 4 below calls the ECRP algorithm to create clusters and find the set of medoid nodes; for every cluster we get the list of source nodes sensing data and we find finally the itineraries of MAs in the network.

**Algorithm 4** The Optimal Multi-agents Itinerary Planning algorithm

**Notation:**

$\boldsymbol{v}$: the number of clusters in the network /*see Equation 9*/
**i**: the index of the cluster
$\boldsymbol{E^i_{itinerary}}$: the energy consumed by the MA to gather data in the cluster i

**Initialisation**
i←1;

**Begin**
1: Find the set of the medoid nodes $MN$ and create the clusters with ECRP algorithm;
2: **While** i<=$v$ **do**
3: List_source$_i$←Find the list of the source nodes of the cluster i with the Equation 10;
4: $IP_i$←Itinerary ($MN_i$,List_source$_i$); /*see algorithm3*/
5: i←i+1;
6: **End While**
**End**

## 5. Simulation and performances

### 5.1 Simulation setting

We choose to implement our proposed algorithm in a network of 800 nodes randomly deployed in a sensed area of 500m×1000m. The sink node is located on the left side of the area 55m away from the nearest node and the location of the source nodes will be determined by the equation 10. The total number of source nodes ($N_s$) in the network changes during the simulation from 10 to 40; the energy consumed by the nodes for reception and transmission is respectively 10 and 50 nj/bit.
The OMIP algorithm will be implemented and compared against four existing algorithms of MIP: CL-MIP, DSG-MIP, BST-MIP and ILS. The ns2 simulator tool [34] will be used to evaluate the performance of these algorithms, the balancing factor α for BST-MIP algorithm is set to 0.6 and for DSG-MIP the value of the threshold angle gap is set to π/6. The parameters of the simulation are shown in the table below:

**Table 2.** Simulation parameters

| Parameters | Values |
|---|---|
| Node distribution | Random |
| Size of the network($m^2$) | 1000×500 |
| Initial energy of the sensors | 5 Joules |
| Total number of sensor nodes | 800 |
| Raw data size | 2KB |
| Processing code size | 1KB |
| MA accessing delay | 10ms |
| Radio transmission range | 60m |
| Aggregation ratio($\rho$) | 90% |
| Reduction ratio($\Omega$) | 80% |
| Critical remaining energy | 0.01J |
| Data processing rate | 50 Mbps |
| Bandwidth of the network | 0.25Mbps |
| Mac layer standard | 802.15.4 |

### 5.2 Performance metrics

In order to evaluate the performance of the MIP algorithms we take into account the following metrics:

- Energy consumption: it's the average of energy consumed by the MAs to gather data of the network, including the energy consumed by the source nodes to transmit, receive, retransmit, process and aggregate data; in addition to the energy consumed by the intermediate nodes.
- End to end delay: the data gathering operation by the MA takes time to accomplish in every cluster. In SIP this time is the average period taken by the sink to dispatch the MA and to recover it with the gathering data. In MIP, the multiple agents circulate in the network in parallel, for that the delay of the last MA arriving to the sink is considered to be the gathering task duration.
- Energy-delay product (EDP): it shows the effectiveness of every MIP algorithm in terms of both energy consumption and end to end delay. As the value of EDP increases, the performance indicator of the algorithm drops. For real-time application in WSN, the time is a constraint along with a limited resource of energy. The integrated performance EDP is equal to the multiplication of the previous metrics: energy and delay.
- Accumulated hop count: the MAs dispatched by the sink move among the source nodes of the network to gather data. Along the route, they will possibly cross intermediate nodes; thus, the accumulated hop count is the total number of sensor nodes crossed by the mobile agents during the gathering operation. To enhance the tradeoff between energy consumption and end to end delay we optimize the itineraries of the MAs by minimizing the hop count.

### 5.3 Performance comparison

The evolution of the energy consumption according to the number of source nodes is represented for the five MIP algorithms in the Figure 8, the BST-MIP consumes more energy than the other algorithms and is the most demanding in terms of energy consumption. The three algorithms: CL-MIP, DSG-MIP and ILS yield the same value in the beginning of the simulation but the gap between them expands while the number of source nodes is growing. The difference reaches 0.06J/Task between CL-MIP and ILS at $N_s$=40. OMIP is the best in terms of energy consumption; it stays almost stable until $N_s$=15, beyond this value it grows rapidly and reaches 0.35J/Task at $N_s$= 40, but stays far bellow BST-MIP (energy saving of 0.2J/Task).

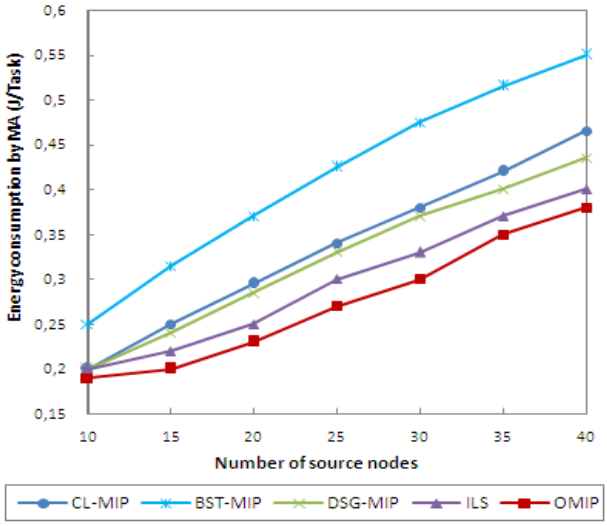


**Figure 8.** Energy consumption by mobile agent in terms of the number of source nodes in the network

In Figure 9, the three algorithms BST-MIP, ILS and OMIP give the best performance in term of end to end delay. Their respective delays grow with the increasing number of source nodes. CL-MIP gives the worst result due to minimal number of MA deployed in the network in comparison with the other MIP algorithms. The difference of the delay between CL-MIP and OMIP is 0.1s at $N_s$=10 and grows up to 0.4s when $N_s$=40.

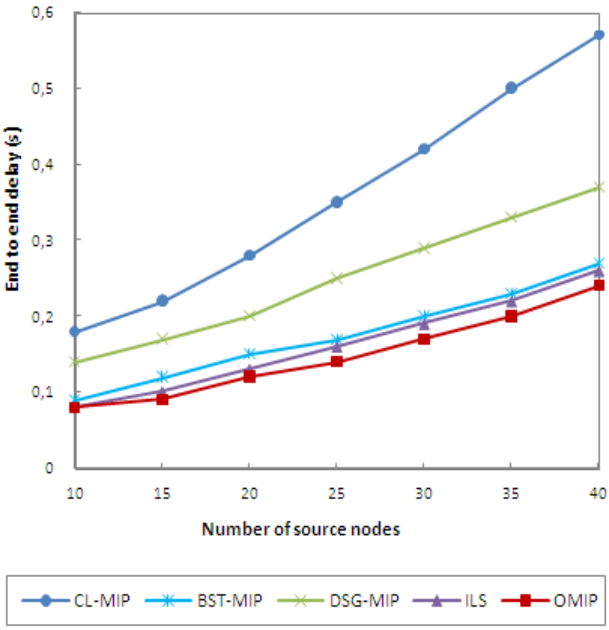


**Figure 9.** End to end delay in terms of the number of source nodes in the network

The integrated performance EDP is shown in Figure 10; CL-MIP is the least efficient algorithm with the highest values of EDP, due to the large values of the delay. When $N_s$=40, its EDP value exceeds twice that of BST-MIP and triple of ILS and OMIP. The BST-MIP and SDG-MIP are close and have almost the same values of EDP when $N_s$ is less than 20. The same thing for ILS and OMIP schemes, they begin with almost the same values, and with the increasing number of source nodes OMIP progressively shows its efficiency against ILS.

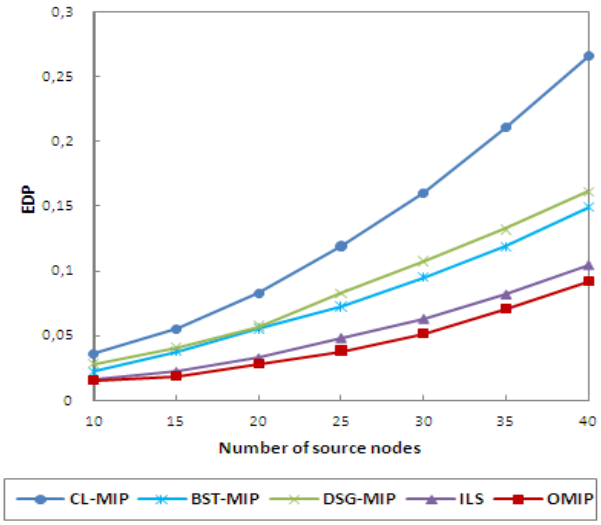


**Figure 10.** Energy-delay product in terms of the number of source nodes in the network

The accumulated hop count is represented for the five MIP algorithms in Figure 11. The MAs move among many nodes to gather data of network; in CL-MIP the hop count is the largest, in contrast with BST-MIP, ILS and OMIP. Their hop count is lower and convergent, which means there are a small number of intermediate nodes crossed by the MAs in their travel between source nodes.

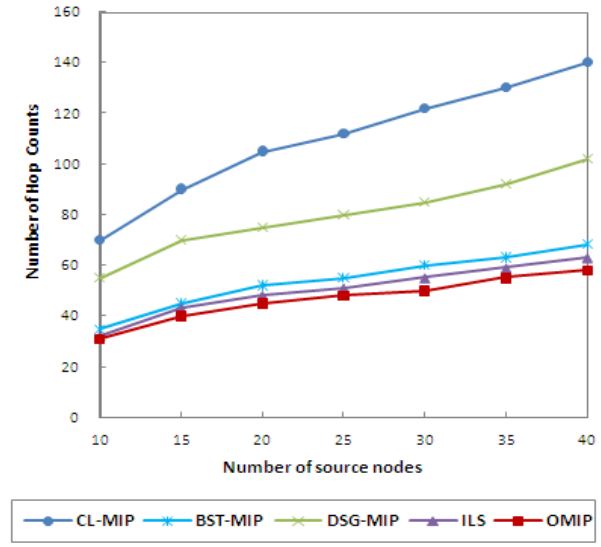


**Figure 11.** Accumulated hop count in terms of the number of source nodes in the network

## 6. Conclusion and perspectives

The approach of itinerary planning with multiple MAs is more efficient than the traditional approaches (i.e. client/server and single MA architectures). With the OMIP algorithm it's possible to divide the sensed field into disjoint and balanced groups using ERCP algorithm and assign an agent for every cluster. This latter works simultaneously with its peers, throughout the network to gather, process and aggregate data.

The MIP algorithm presented in this article takes into account the transmission range of the nodes in the creation of the itineraries. In some cases, the MA has to traverse some intermediate nodes to reach the next source node, then the itinerary of the MA is related to the path followed among the source nodes and the intermediate nodes; the OMIP follows a good policy of routing, by minimizing the number of source nodes and intermediate nodes in the itinerary of the MA without any loss of information. Consequently, it improves the performance metrics by extending the lifetime of the network, and reducing the accumulated hop count and the end to end delay. The simulations have proved the efficiency of our algorithm, in terms of these performance metrics against the existing MIP algorithms.

In Our future work we plan to divide the network into honeycomb clusters and change the position of the cluster-head cell to balance the energy consumption in intra-clusters and prolong the lifetime of the network.

## References

[1] M. Alipour, K. Faez, "On design of mobile agent routing algorithm for information gain maximization in wireless sensor networks," In Proceedings of the Sixth International Conference on systems and Networks Communication (ICSNC'2011), Barcelona, Spain, pp. 193–198, 2011.

[2] J. Wang, Y. Zhang, Z. Cheng, X. Zhu, "EMIP: energy-efficient itinerary planning for multiple mobile agents in wireless sensor networks," Telecommun Syst, Vol. 62, No. 1, pp. 93–100, 2016.

[3] D. Gavalas, I.E. Venetis, C. Konstantopoulos, G. Pantziou, "Energy-efficient multiple itinerary planning for mobile agents-based data aggregation in wireless sensor networks," Telecommun Syst, Vol. 63, No. 4, pp. 531–545, 2016.

[4] D. Gavalas, I.E. Venetis, C. Konstantopoulos, G. Pantziou, "Mobile Agent Itinerary Planning for WSN Data Fusion: Considering Multiple Sinks and Heterogeneous Networks," International Journal of Communication Systems, Vol. 30, No. 8, 2017.

[5] H. Jamal A. Nasir, K.R. Ku-Mahamud, E. Kamioka3, "Enhanced Ant-Based Routing for Improving Performance of Wireless Sensor Network," International Journal of Communication Networks and Information Security, Vol. 09, No. 3, 2017.

[6] H. Qi, F. Wang, "Optimal itinerary analysis for mobile agents in ad hoc wireless sensor networks," In Proceedings of the 13th IEEE International Conference on Wireless Communications, Calgary, Canada, pp. 147–153, 2001.

[7] M. Chen, L.T. Yang, T. Kwon, L. Zhou, M. Jo, "Itinerary planning for energy-efficient agent communications in wireless sensor networks," IEEE Transactions on Vehicular Technology, Vol. 60, No. 7, pp. 3290–3299, 2011.

[8] M. Chen, S. Gonzalez, Y. Zhang, V.C.M. Leung, "Multi-agent itinerary planning for wireless sensor networks," In Proceedings of the IEEE International Conference on Heterogeneous Networking for Quality, Reliability, Security, and Robustness (QShine'2009), Las Palmas de Gran Canaria, Spain, pp. 584–597, 2009.

[9] M. Chen, W. Cai, S. Gonzalez, V.C. Leung, "Balanced itinerary planning for multiple mobile agents in wireless sensor networks," In Proceedings of the Second International Conference of on Ad Hoc Networks (ADHOCNETS'2010), Victoria, Canada, 2010.

[10] D. Gavalas, A. Mpitziopoulos, G. Pantziou, C. Konstantopoulos, "An approach for near optimal distributed data fusion in wireless sensor networks," Journal of Wireless Networks, Vol. 16, No. 5, pp. 1407–1425, 2009.

[11] A. Mpitziopoulos, D. Gavalas, C. konstantopoulos, G. pantziou, "CBID: a scalable method for distributed data aggregation in WSNs," International Journal of Distributed Sensor Networks, Vol. 2010, No. 1, 2010.

[12] C. Konstantopoulos, A. Mpitziopoulos, D. Gavalas, G. pantziou, (2010). "Effective determination of mobile agents Itineraries for data fusion tasks on sensor networks," IEEE Transactions on Knowledge and Data Engineering, Vol. 22, No. 12, pp. 1679–1693, 2010.

[13] M. Cai, M. Chen, T. Hara, L. Shu, "GA-MIP: genetic algorithm based multiple mobile agents itinerary planning in wireless sensor networks," In Proceedings of the Fifth International Wireless Internet Conference (WINCON'2010), Singapore, 2010.

[14] I. Aloui, O. Kazar, L. Kahloul, S. Servigne, "A new Itinerary Planning Approach Among Multiple Mobile Agents in Wireless Sensor Networks (WSN) to Reduce Energy Consumption," International Journal of Communication Networks and Information Security, Vol. 07, No. 2, 2015.

[15] W.R. Heinzelman, A. Chandrakasan, H. Blalakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences. Hawaii, USA, 2000.

[16] W.R. Heinzelman, A. Chandrakasan, H. Blalakrishnan, "An application-specific protocol architecture for wireless micro sensor networks," IEEE Transactions on Wireless Communication, Vol. 1, No. 4, pp. 660–670, 2002.

[17] C.L. Fok, G.C. Roman, C. Lu, "Agilla: a mobile agent middleware for self adaptive wireless sensor networks," ACM Transactions on Autonomous and Adaptive Systems, Vol. 4, No. 3, 2009.

[18] TinyOS, http://www.tinyos.net.

[19] Y. Kwon, S. Sundresh, K. Mechitov, G. Agha, "ActorNet: an actor plateform for wireless sensor networks," In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'2006), Hakodate, Japan, 2006.

[20] Y. Kwon, K. Mechitov, G. Agha, "Design and implementation of a mobile actor platform for wireless sensor networks," Concurrent Objects and Beyond, Springer, Berlin, pp. 276–316, 2014.

[21] Sun Small Programmable Object Technology (Sun Spot), http://www.sunspotworld.com.

[22] F. Aiello, G. Fortino, R. Gravina, A. Guerrieri, "A java-based agent platform for programming wireless sensor networks," The computer journal, Vol. 54, No. 3, pp. 439–454, 2011.

[23] F. Aiello, G. Fortino, S. Galzarano, R. Gravina, A. Guerrieri, "An analysis of java-based mobile agent platforms for wireless sensor networks," Multiagent and Grid Systems, Vol. 7, No. 6, pp. 243–267, 2011.

[24] F. Aiello, A. Carbone, G. Fortino, S. Galzarano, "Java-based mobile agent platforms for wireless sensor networks," In Proceedings of the International Multiconference on Computer Science and Information Technology, Wisla, Italy, pp. 165–172, 2010.

[25] G. Giuseppe, M. Conti, M.D. Francesco, A. Passarella, "Energy conservation in wireless sensor networks: a survey," Ad Hoc Networks, Vol. 7, No. 3, pp. 537–568, 2009.

[26] X. Wang, M. Chen, T. Kwon, H. C. Chao, "Multiple mobile agents itinerary planning in wireless sensor networks: survey and evaluation," IET Communications, Vol. 5, No. 12, pp. 1769–1776, 2011.

[27] R. Amine, B. Khalid, O. Mohammed, "Efficient hierarchical clustering routing in wireless sensor networks," International Journal of Future Generation Communication and Networking, Vol. 9, No. 9, pp. 109–118, 2016.

[28] A. Boukerche, H.A.B.F. Oliveira, E.F. Nakamura, A.F.A. Loureiro, "Localization systems for wireless sensor networks," IEEE Wireless Communications, Vol. 14, No. 6, pp. 6–12, 2007.

[29] Z. Fang, Z. Zhao, X. Cui, D. Geng, L. Du, C. Pang, "Localization in wireless sensor networks with known coordinate database," EURASIP Journal on Wireless Communications and Networking, Vol. 2010, No. 2, 2010.

[30] R. Misra, S. Shukla, V. Chandel, "Lightweight localization using trilateration for sensor networks," International Journal of Wireless Information Networks, Vol. 21, No. 2, pp. 89–100, 2014.

[31] J. Xu, W. Liu, F. Lang, Y. Zhang, C. Wang, "Distance measurement model based on RSSI in WSN," Wireless Sensor Network, Vol. 2, No. 8, pp. 606–611, 2010.

[32] R.T. Ng, J. Han, "Efficient and effective clustering methods for spatial data mining," In Proceedings of the International Conference on Very Large Data Bases, Santiago, Chile, pp. 144–155, 1994.

[33] R.T. Ng, J. Han, "CLARANS: a method for clustering objects for spatial data mining," IEEE Transactions on Knowledge and Data Engineering, Vol. 14, No. 5, pp. 1003–1016, 2002.

[34] The Network Simulator (ns2), http://www.isi.edu/nsnam/ns.

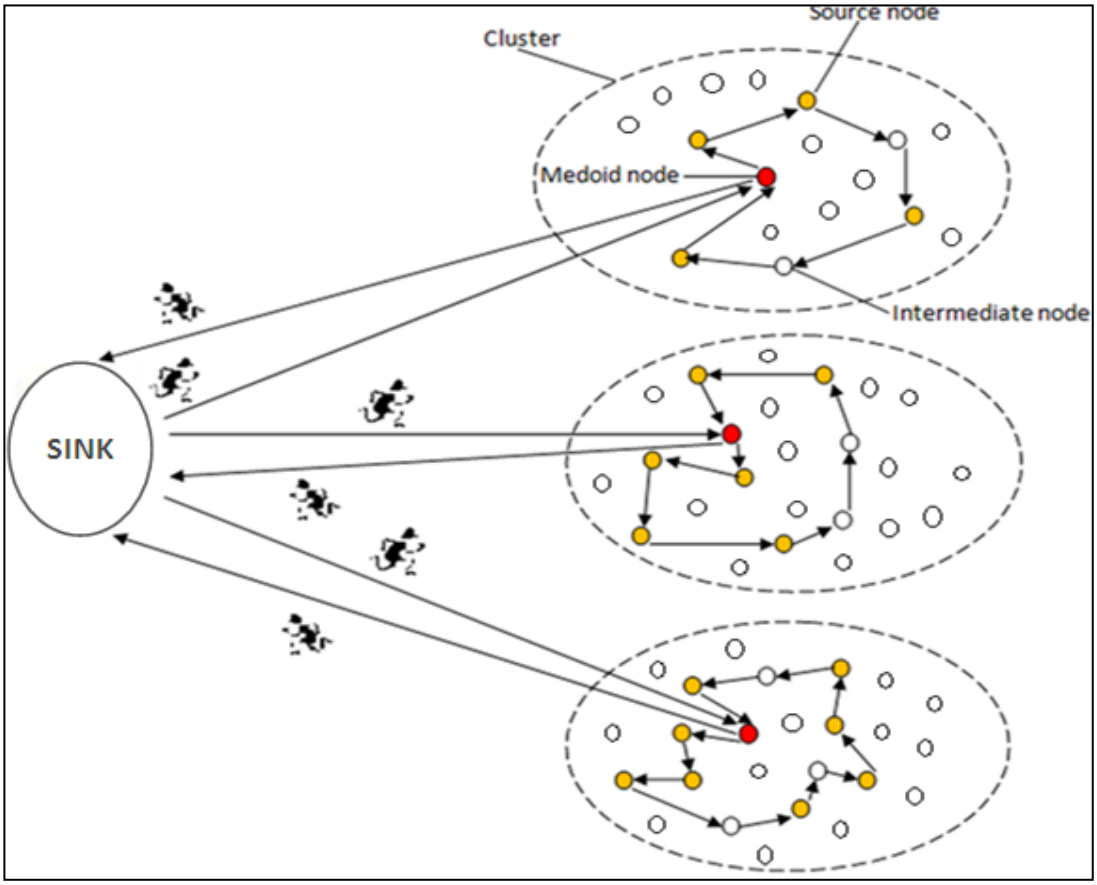


**Figure 4.** Architecture of our network