An Hierarchical Scheduled Algorithm for Data Dissemination in a Brown Planthopper Surveillance Network

Bao Hoai Lam[1(✉)], Tuyen Phong Truong[1], Ky Minh Nguyen[2], Hiep Xuan Huynh[3], and Bernard Pottier[1]

[1] Université de Bretagne Occidentale, LAB-STICC, UMR CNRS 6285, Brest, France
{bao-hoai.lam,tuyen-phong.truong,pottier}@univ-brest.fr
[2] Can Tho University of Technology, Can Tho, Vietnam
nmky@ctuet.edu.vn
[3] DREAM Team/UMI 209 UMMISCO-IRD, Can Tho University, Can Tho, Vietnam
hxhiep@ctu.edu.vn

Abstract. The paper proposes a new approach to model the hierarchical structure and implement a scheduled algorithm for disseminating data of a brown planthopper surveillance network based on Wireless Sensor Network (WSN) approach. In the hierarchical model, light trap sensor nodes in the same province compose a sub network at level 1 while sink nodes of these networks compose another sub network at level 0. Thanks to this structure, there are 2 types of data dissemination: local at sub networks at level 1 and cluster at sub network at level 0. These behaviors are monitored by a timer which calculates the next execution time for each node. The model and its algorithms are simulated using data collections of the brown planthopper surveillance network in Mekong Delta, especially in Cantho, Angiang, Dongthap - 3 typical rice provinces in the delta. Structure of the model is suitable for the hierarchical management policies of a light trap network in a large region and the role of WSN in collecting data is emerged via the simulation.

Keywords: BPHSUN · Brown planthopper surveillance network · BPH · WSN · Timer

1 Introduction

Light trap network is one of effective solutions for insect pest management in a large area. For example, Rothamsted light trap networks [1,2] have been established in Britain since the 1960s to monitor insect migrations and populations in order to provide warning systems. Another example is that a light trap network in Mekong Delta [7] with more than 340 light traps can provide hopper densities and thanks to data collection form the network, people may know what types of insects are there in their fields and if they are in a controllable level.

P.C. Vinh et al. (Eds.): ICTCC 2016, LNICST 168, pp. 246–263, 2016.
DOI: 10.1007/978-3-319-46909-6_23

The idea of light trap network based on Wireless Sensor Network (WSN) approach [9] emerges as a suitable choice for insect, particularly Brown Planthopper (BPH) monitoring since they can help monitoring hopper behaviors automatically as well as providing meteorological factors. This kind of solution use sensors in each trap to sense surrounding conditions and a whole network becomes a massive coordinated sensing machine. These pieces of sensed data are collected and sent to a data center via a wireless network for post processing. This solution can be shortly called as BPH surveillance network.

Due to a large area of BPH surveillance network distribution, it is necessary to maintain an appropriate light trap network topology in order to collect data. This topology illustrates the topology of the observation WSN and influences on algorithms for collecting data from the WSN as well. Moreover, management policies of the surveillance network also rely on this topology such as: authorization of light traps, decision making.

This paper proposes a new hierarchical structure to model a BPH surveillance network topology as well as describes an appropriate scheduled algorithm for data dissemination in the network. In this work, the light trap sensor network is considered as a multi-level graph in which each light trap sensor node is modelled as a node and 2 sensor nodes located in their communication ranges, the maximum distance that radio signal from a node can be reached, can be considered as candidates to establish an edge. Based on the multi-level graph, an hierarchical structure with multi-level sub networks, along with a scheduled algorithm, is presented for data dissemination inside the BPH surveillance network in a large region.

The structure of this paper is as follows. Section 2 summarizes some previous work relating to insect surveillance network topology. Next section depicts the topology of a BPH surveillance network as an hierarchical graph model. Section 4 is about data behaviors of the BPH surveillance network including scheduled algorithms for sending and receiving collected data from sensor nodes. Next section illustrates some simulation results for data behaviors of the surveillance network with data collection in Cantho, Angiang and Dongthap. The last section is our conclusion and future plans.

2 Related Work

Light trap method is one of solutions to prevent high densities of spruce budworm in Canadian forests [3,4]. This method allows people to participate insect trapping by giving light traps to them and track their traps from June to end of August every year. Periodically, people only report estimated densities of insects via a website, an application or even with a paper and a pen. Finally, trap samples are collected and counted in a lab environment. Applications of data collections from these light traps are variant, for example, thanks to wing wear and body size measurements of adult spruce budworms captured at light traps in some previous years, some useful inference on seasonal patterns related

to reproduction can be archived [5]. However, these light traps seem not to compose a network, instead, they create a combination of traps to collect data for post processing.

Rothamsted light trap networks [1,2] have been established in Britain since the early 1960 s to monitor pest insects. The initial ideas of these networks is to understand the population change when pest insects occur. By the time passing, these networks have also been used to monitor the responses of insects to climate and environmental change and as well as to measure and analyze biodiversity [18, 19]. Nevertheless, few information about topologies of these networks is found.

If a light trap network is based on wireless sensor network (WSN) approach (as proposed in [8,9]), its topology is identified by the observation WSN topology. Although these pieces of above work use mesh network to distribute data, there is no information about how data is transmitted via the network topology.

In addition, there are not many investigation of hierarchical structure in modeling insect surveillance network topology.

3 Multilevel Graph of the BPH Surveillance Network

3.1 Description

The BPH surveillance network in a large region depicts an hierarchical structure with 3 levels (Fig. 1). Level 2 illustrates light trap sensor nodes to collect data. These sensor nodes compose sub networks at level 1 and each sub network is represented by a sink node where data in the sub network is assembled. These sink nodes constitute another sub network at level 0 with a gateway where collected data from the whole network is aggregated for post processing.

3.2 Sub Network at Level 1

A sub network at level 1 (small circles in Fig. 1) can be considered as a graph $G_i = (V_i, E_i)$ where V_i is a collection of automatic light trap sensor nodes in the same province i and E_i is a collection of edges. An edge between 2 nodes in V_i is composed if the distance between them is at most the transmission range, the maximum distance that a single transmission of a node can be received by all nodes in its vicinity. All sensor nodes are assumed to have the same transmission range.

In the graph G_i, a special node is elected as a sink (leader) where data is assembled and stored. In this case, sensor nodes have abilities to sense surrounding conditions (environmental factors and hopper densities) and transmit sensed data to its neighbors. However, the sink node can only receive data from its neighbors in the same sub network at level 1.

3.3 Sub Network at Level 0

Sink nodes of all sub networks at level 1 can be considered as a graph $G = (V, E)$ where V is a collection of sink nodes, nodes concentrating data from sub networks

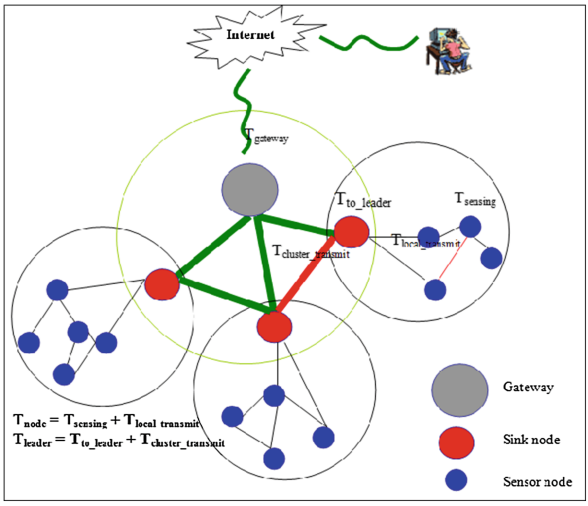

Fig. 1. Hierarchical structure of hopper observation WSN.

at level 1, and E is a set of edges. An edge is created between 2 nodes if the distance between them is at most the transmission range. Similar to sub networks at level 1, all sink nodes have the same transmission range.

In this sub network, each sink node can transmit aggregated data, data accumulated from collected data in the corresponding sub network at level 1, to its neighbors. There is no sensing activity in the sub network at level 0.

A special node is elected as a gateway. This node can be a sink or a new node to store the final data. The main task of the gateway is to receive the aggregated data from sinks and there could be an application to which connects for decision making.

3.4 BPHSUN

The composition of the sub network at level 0 and sub networks at level 1 depicts an hierarchical structure for the BPH surveillance network. Therefore, it can be called as **B**rown **P**lanthopper Hierarchical **SU**rveillance **N**etwork (BPHSUN).

4 Data Transmission in the BPHSUN

4.1 Data Packet

Data is packed as packets and transmitted through the hierarchical structure of the BPHSUN. In the BPHSUN, there are 2 types of packets:

– Local packet: data sensed at each sensor node is packed as a local packet.
– Cluster packet: a packet aggregated at each sink node.

The structure of a local packet is depicted as in Fig. 2. In this structure, ID is the unique ID of the packet and location ID represents the ID of the sensor node that senses environmental factors. The time that the sensor node senses data is described by a time stamp. Source ID depicts the sensor node that sends the packet while destination ID is the ID of next node that the packet is received. Normally, the destination ID is identified thanks to a routing table, a table that routes packets to a sink (or a gateway). These attributes compose a header of the packet. In addition to the header, the packet contains a data part which stores surrounding conditions such as light intensity, temperature, humidity, wind velocity, wind direction and hopper density.

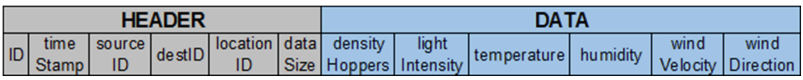

Fig. 2. Local packet in the BPHSUN.

The structure of a cluster packet is almost similar to that of a local packet. However, the data part of the cluster packet contains min, mean, max values of environmental factors after a period of time.

A packet here depicts a structure to maintain a piece of spatial-temporal data. Indeed, time stamp illustrates the temporal aspect while location id describes the spatial one and the data part of the packet becomes data aspect. Figure 3 is an example of this piece of data. It can be translated as: at the time *01/02/2016 08:07:56 AM*, the sensor node *10* has *500 hoppers caught* at the temperature 29°C and *5.5* km/h wind velocity.

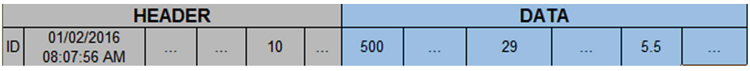

Fig. 3. Example of a piece of spatial-temporal data.

Consequently, a local packet can be declared as the following C based pseudo code:

```
typedef struct{
        int ID;
        int sourceID, desID;
        int location;// local ID of sensor node that senses data
        int timeStamp; //time stamp of sensing
        int dataSize;
        float densityHoppers; // density of hoppers
        float lightIntensity; // light intensity
        float temperature, humidity; // temperature & humidity
        float velocityWind;      // wind velocity
        float directionWind;     // wind direction
}LocalPacket;
```

The declaration of a cluster packet is similar to that of a local packet. The difference is that the cluster packet may contain max, mean, min values of surrounding conditions as well as hopper densities.

4.2 Packet Dissemination

There are 2 procedures to disseminate data in the BPHSUN based on types of packets: local packet and cluster packet procedures.

Local packet dissemination. Local packet dissemination illustrates behaviors of a sensor node in each sub network at level 1. These behaviors compose of sensing surrounding conditions, packing into a local packet and sending the packet to a sink node (Fig. 4).

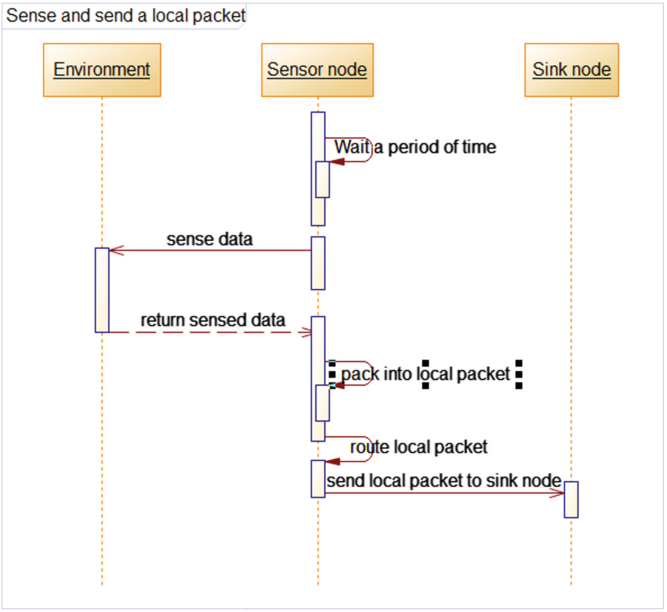

Fig. 4. Procedure of sensing and sending a local packet

Figure 4 depicts behaviors of a sensor node of each sub network at level 1. First, next execution time of a sensor node is identified by a timer. Next, the sensor node measures surrounding conditions and receives sensed data from environment, then these pieces of data are packed as a local packet (as in Fig. 2). A pre-calculated routing table is used to route the local packet to its sink node where sensed data is aggregated periodically.

Cluster packet dissemination. Cluster packet dissemination describes behaviors of a node in the sub network at level 0. These behaviors represent a procedure to aggregate packets collected at each sensor node at level 1, pack into a cluster packet and send to a gateway.

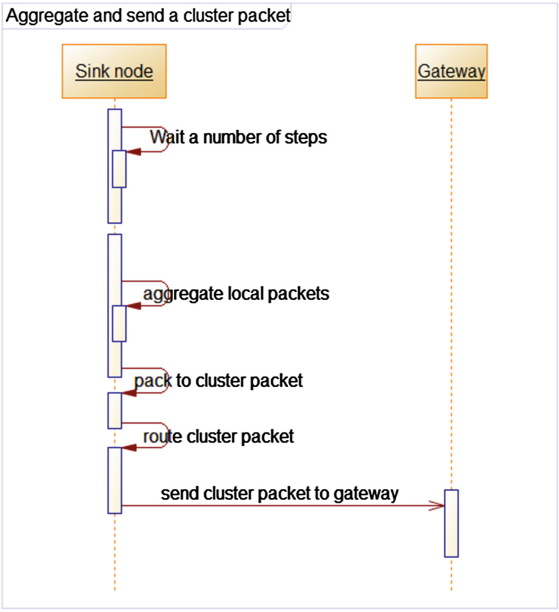

Fig. 5. Procedure of sensing and sending a cluster packet.

Figure 5 illustrates behaviors of a node in the sub network at level 0. Initially, a timer identifies the time for the next execution of the node. Next, the node aggregates local packets collected from each sensor node (in the same sub network at level 1 that the node plays as a sink) after the period between 2 adjacent executions of the node, then these pieces of data are packed as a cluster packet (similar to Fig. 2). A pre-calculated routing table is used to route the cluster packet to the node's neighbors, then to the gateway.

4.3 Timer

In the BPHSUN, timer is a mechanism calculating the time when nodes (sensor, sink and gateway) execute their tasks. This time can be figured out based on sensing times of sensor nodes and communication times to transmit packets to the gateway.

Assume that each sensor node contains n sensors, t_i is a duration of time to read sensor i. Therefore, $T_{read} = \{t_1, t_2, ..., t_n\}$ is a collections of times to read sensors in a sensor node.

Let $T_{sensing}(s)$ is the sensing time, a duration of time that a sensor node finish reading all its sensors. This time is calculated as:

$$\Rightarrow T_{sensing} = Max\{t_1, t_2, ..., t_n\} = Max\{T\} \tag{1}$$

To transmit data to the gateway, following factors are considered:

- f: the frequency used to transmit data (Hz).
- LocalDataSize, ClusterDataSize: sizes of a local packet and a cluster packet, respectively (bit).
- DataRate: the rate (speed) to transmit data ($Kbps$). This rate depends on the frequency used to transmit data.

The communication time $T_{local\_transmit}$ from one node to another node in a sub network at level 1 is shown as below:

$$T_{local\_transmit} = \frac{LocalDataSize}{DataRate} \tag{2}$$

Similarly, the communication time $T_{cluster\_transmit}$ in the sub network at level 0 is measured as:

$$T_{cluster\_transmit} = \frac{ClusterDataSize}{DataRate} \tag{3}$$

Therefore, the duration of time that a sensor finish executing its tasks (Fig. 1) is calculated as:

$$T_{node} = T_{sensing} + T_{local\_transmit} \tag{4}$$

Let T_{to_leader} be a duration of time that a leader (sink node) finish collecting data from all other nodes (Fig. 1), then T_{to_leader} is estimated as:

$$T_{node} \leq T_{to\_leader} \leq nT_{node} \tag{5}$$

where n is the number of sensor nodes in a sub network at level 1. Indeed, in the best case, all sensor nodes of a sub network at level 1 transmit its data directly to a sink node at the same time, then T_{to_leader} is approximately T_{node}. However, in the worst case, sensor nodes perform sequentially, then T_{to_leader} is around nT_{node}.

Assume that there are m sink nodes in the sub network at level 0. Let T_{leader} be a time that a sink node collects data from its sensor nodes at leaf level and sends these pieces of data to a neighbor of it in the sub network at level 0 (Fig. 1). Another assumption is that $T_{gateway}$ is a total of time that a packet reaches the gateway. Then:

$$T_{leader} = T_{to\_leader} + T_{cluster\_transmit} \tag{6}$$

$$T_{leader} \leq T_{gateway} \leq mT_{leader} \tag{7}$$

The interval time T between 2 next actions of a sensor node in the BPHSUN can be calculated as the maximum of $T_{gateway}$ after adding an error-time Δt. Therefore, T is estimated as:

$$T = mT_{leader} + \Delta t \tag{8}$$

$$\Rightarrow T = m(n(T_{sensing} + T_{local\_transmit}) + T_{cluster\_transmit}) + \Delta t \quad (9)$$

The above interval time T is used for nodes of sub networks at level 1. However, sink nodes do not forward data sensed from sensor nodes to the gateway immediately, instead, they wait some working steps of sensor nodes in order to aggregate data. Thus, after a multiple of T time unit, nodes at level 0 aggregate data and transmit to the gateway.

4.4 Routing Table

Routing table [20] is a structure to store shortest paths from a node to other nodes in a WSN. In the BPHSUN, because automatic light trap sensor nodes located at fixed positions, routing table is prev-calculated and stored to disseminate data to sinks or the gateway.

Distance vector algorithm [20] is used to calculate routing tables for a WSN $G(V, E)$. The algorithm is shown as below:

ALGORITHM. Calculate a routing table for node v to other nodes in $G(V, E)$

INPUT: c(v, w): the direct cost from v to w ($w \in V$).

OUTPUT: D(v, w): distance between v and w ($w \in V$). Next(v, w): next node to reach to w from v. ($w \in V$)

```
createRoutingtable(){
  for each (w in V)
        D[v, w] = 0;
  D[v, v] = _INFINITY;
  // Find route from v to others
  for each (w in V){
        if (w == v) continue;
        /* Select the shortest distance from v to its neighbors
                after adding direct costs */
        minCost = D[v, w]; // old distance
        neighbor = -1;
        nextNode = Next(v, w);
        for each (j in v.getNeighbors()){
                neighbor = j;
                newCost = c[v,j] + D[j, w];
                if (newCost < minCost){
                        minCost=newCost;
                        nextNode=neighbor;
                }
        }
        // Update routing table
        D[v,w]=minCost;
        Next[v,w]=nextNode;
  }
}
```

The above algorithm is executed parallely for all nodes $v \in V$, then routing tables for all node $v \in V$ to other nodes are found after 1 execution step. This procedure of calculating routing tables for all nodes is looped until distances are unchanged.

Example. Let a WSN given by a graph $G(V, E)$ (Fig. 6). Assume that the direct cost from a node to its neighbors is 1.

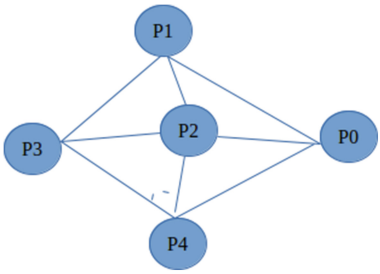

Fig. 6. Example of a graph to calculate routing tables.

The following result are routing tables of all nodes in the graph G after 2 execution steps. After 2 steps, routing tables of all nodes are unchanged, thus, they become final routing tables.

	P0		P1		P2		P3		P4	
	D	*Next*	*D*	*Next*	*D*	*Next*	*D*	*Next*	*D*	*Next*
P0	0	-1	1	0	1	0	2	1	1	0
P1	1	1	0	-1	1	1	1	1	1	1
P2	1	2	1	2	0	-1	1	2	1	2
P3	2	1	1	3	1	3	0	-1	1	3
P4	1	4	1	4	1	4	1	4	0	-1

Fig. 7. Routing tables after 2 steps.

4.5 Routing Local Packets

This procedure takes place in sub networks at level 1. It is a process of sending and receiving packets at each sensor node based on a routing table so that all packets are concentrated at the sink node.

Before sensed data is transmitted, it is packed as a local packet (as Fig. 2). The local packet consists of an ID, source ID, destination ID, location, time stamp and other measured values of environmental factors. The attribute location depicts the local ID of the sensor node that senses these surrounding conditions.

Example: Assume that P1, P2, P3, P4 are sensor nodes and P0 is the sink node of the graph in Fig. 6, according to routing tables in Fig. 7, collected data from P1, P2, P4 can send directly to P0 while P3 needs 2 hops to reach to P0 (P3 - P1 - P0). Figure 8 depicts a round how sensed data from sensor nodes reaches the sink node P0.

HEADER						DATA					
1	...	P1	P0	P1	...	500	...	30	...	...	...
2	...	P2	P0	P2	...	550	...	29	...	...	...
3	...	P3	P1	P3	...	600	...	30	...	...	...
4	...	P4	P0	P4	...	650	...	29.5	...	...	...
5	...	P1	P0	P3	...	600	...	30	...	...	...

Fig. 8. Example of routing local packets.

To implement this mechanism, a sending buffer and a receiving buffer are maintained at each sensor node. These buffers are used in 2 following methods: sendLocalPackets() and receiveLocalPackets() to send and receive local packets at a node v. All nodes in the WSN execute these methods concurrently.

Sending packets. Sending packets takes place at sensor nodes of the WSN after they are granted execution times from the timer. First, environmental factors are sensed by sensors of a node and these pieces of sensed data are packed as a packet. Next, the packet is added to sending buffer of the node. The next step is to move all packets from the node's receiving buffer to its sending buffer in order to send to the gateway thanks to the routing table.

The algorithm is described as followed:

ALGORITHM. send local packets at a sensor node

INPUT: Node v, routing table t, receiving buffer *v.receiveBuff*
OUTPUT: sending buffer *v.sendBuf*

```
sendLocalPacket(){
 if (isExecutionTime()){
   if (!isLeader(v)){
     senseData();       //Sense data from environment
     createPacket(p); //Create a packet
     l = Leader(v);     //Find leader of v

     //Next hop of v is destination ID of p
     p.destinationID = t.Next[v][l];

     v.sendBuff.Add(p);
     for each (packet p in v.receiveBuff)
       v.sendBuff.Add(p);
     v.receiveBuff.Clear();
   }
 }
}
```

Receiving local packets. When sensor nodes are granted execution times, they start receiving packets from their neighbors. Main idea of receiving packets at a node v is to locate in the sending buffer of each neighbor of v in order to find packets considering v as their destination IDs, then these packets are added to the receiving buffer of v.

ALGORITHM. receive local packets at a node

INPUT: Node v, routing table t
OUTPUT: receiving buffer *v.receiveBuf*

```
receiveLocalPacket(){
 if (isExecutionTime()){
   l = Leader(v); //Find leader of v
   for each (Neighbor j of v){
     for each (packet p in j.sendBuf){
       desID = p.destinationID;
       if (desID == v.ID){
         //Next hop of v is destination ID of p
         NextID =  t.Next[v][l];
         p.destinationID = NextID;
         v.receiveBuff.Add(p);
       }
     }
   }
 }
}
```

4.6 Routing Cluster Packets

Routing cluster packets relating to sending and receiving cluster packets at each sink nodes thanks to a routing table so that all cluster packets are concentrated at the gateway. To implement this procedure, each sink node has 2 more buffers: sending and receiving cluster buffers. 2 new buffers are used for transmitting packets in the sub network at level 0. Similar to routing local packets, this procedure is divided into 2 methods: sendClusterPackets() and receiveClusterPackets(). Similar to routing local packets, nodes in the sub network at level 0 execute these above methods parallely.

Sending cluster packets. Sending cluster packets takes place at nodes of the sub network at level 0 when these nodes are granted execution times from the timer. First, each node aggregates data in packets from its sending local buffer grouping by each sensor node after a period of time identified by the timer. After grouping, each group of packets composes a cluster packet and the packet is added to the node's sending cluster buffer.

ALGORITHM. send cluster packets at a node

INPUT: Sink node v, routing table t, sending and receiving buffers of v: v.sendBuf, v.receiveBuf

OUTPUT: sending cluster buffer *v.sendClusterBuf*

```
sendClusterPacket(){
 if (isExecutionTime()){
   if (isLeader(v){
     //Aggregate packets in sending and receiving buffer of v
     //  results are in a temp buffer b
     aggregateData(v.sendBuf, v.receiveBuf, &b);
     for each (cluster packet p in b){
       p.sourceID = v.ID;
       //Next hop of v is destination ID of p
       p.destinationID = t.Next[v][gateway];
       v.sendClusterBuf.Add(p);
     }
   }
 }
}
```

ALGORITHM. receive cluster packets

INPUT: Sink node v, routing table t

OUTPUT: receiving cluster buffer *v.receiveClusterBuf*

```
receiveClusterPacket(){
 if (isExecutionTime()){
   if (isLeader(v) || isGateway(v)){
     for each (neighbor j of v){
       for each (packet p in j.sendClusterBuf){
         desID = p.destinationID;
         if (desID == v.ID){
           // //Next hop of v is destination ID of p
           NextID = t.Next[v][gateway];
           p.destinationID = NextID;
           v.receiveClusterBuf.Add(p);
         }
       }
     }
   }
 }
}
```

Receiving cluster packets. Similar to receiving local packets, the main idea of this algorithm is to locate in the cluster sending buffer of each neighbor of a node v in order to find cluster packets considering v as their destination IDs, then these packets are added to the receiving cluster buffer of v. The algorithm is shown as below:

5 Experiment

5.1 Data Used

The experiment uses the data collection of the light trap network in Mekong Delta by selecting some typical rice provinces to simulate. It uses 4 current light traps in Cantho, 3 in Angiang and 3 in DongThap as sensor nodes. Thus, the BPHSUN contains 3 sub networks at level 1 and 1 sub network at level 0 (Fig. 9). The sub network in Cantho contains sensor nodes P00, P01, P02, P03, P04 which P00 is a sink node. In Angiang, the sub network consists of 4 sensor nodes P05, P06, P07, P08 which P05 is a sink node. Similarly, P09, P10, P11, P12 compose a sub network of Dongthap which P09 is a sink node. At level 0, P00, P05, P09 and the gateway P13 compose another sub network at level 0.

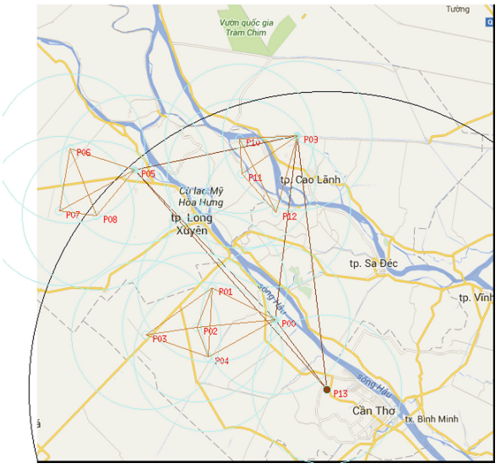

Fig. 9. The BPHSUN composed by light trap sensor nodes in Cantho, Angiang, Dongthap.

5.2 Tools Used

The map including Cantho, Angiang and Dongthap is processed by the tool PickCell in the framework NetGen [12]. Behaviors of the BPHSYN are implemented in CUDA to run the simulation on the NVIDIA card GeForce GTX 680 1.15 GHz with 1536 CUDA Cores (8 Multiprocessors x 192 CUDA Cores/MP).

5.3 Sensor Node Interval Time Calculation

According to Formula 9 the sensor node interval time of the BPHSUN depends on sensing times and communication times.

Actually, sensing time of a sensor is the response time of that sensor. This time relies on type of sensors as well as concrete surrounding conditions. A sensor node here is an automatic light trap [21] consists of following sensors shown in Fig. 10. This figure also depicts the response time of each sensor [14–16].

Sensor	Response time (s)
Wind sensor WindSonic M	0.25
Temperature and humidity sensor DT22	20
Light sensor BH1750FVI	Approximately human eye response

Fig. 10. Response times of sensors in a light trap sensor node

Therefore, the sensing time is $T_{sensing} = 20\,\text{s}$.

LORA technology [17] is used to transmit data in the BPH surveillance network in Mekong Delta [21]. The board Semtech SX1276 [17] is used since it is suitable for allowance frequencies in Vietnam. The specification of this board shows that it has 0.018–37.5 kbps data rate, the frequency 433 MHz. Assume that the board is configured to work with 11 kbps data rate.

According to 4.1, the size of each packet is 48 bytes ($LocalDataSize = 48$). In this simulation, a cluster packet aggregates mean values of environmental factors, therefore, the structure of a cluster packet is similar to that of a local packet. Consequently, $ClusterDataSize = LocalDataSize = 48$ bytes.

$$\Rightarrow T_{cluster\_transmit} = T_{local\_transmit} = \frac{LocalDataSize}{DataRate} = \frac{48 * 8}{11 * 1000} = 0.035\,\text{s}$$

According to formula 9, the interval time T is calculated as:

$$\Rightarrow T = 60.5 + \Delta t(s)$$

Because environmental factors do not change so much during a short period of time, $\Delta t = 1739.5\,\text{s}$ is chosen $\Rightarrow T = 1800\,\text{s} = 30\,\text{min}$. Thus, the interval time between 2 adjacent actions of a sensor node is 30 min. If a light trap works 4 h every night, the sensor node senses environment and transmits data 9 times.

5.4 Scenario *Data Collection After 30 min*

In this scenario, assume that sink nodes aggregate data every 30 min, it means that these pieces of data are calculated after every 2 adjacent steps of a sensor node.

Figure 11 depicts a status of the BPHSUN when sensor nodes and sink nodes execute their tasks. The left of this figure is captured when the sensor nodes are

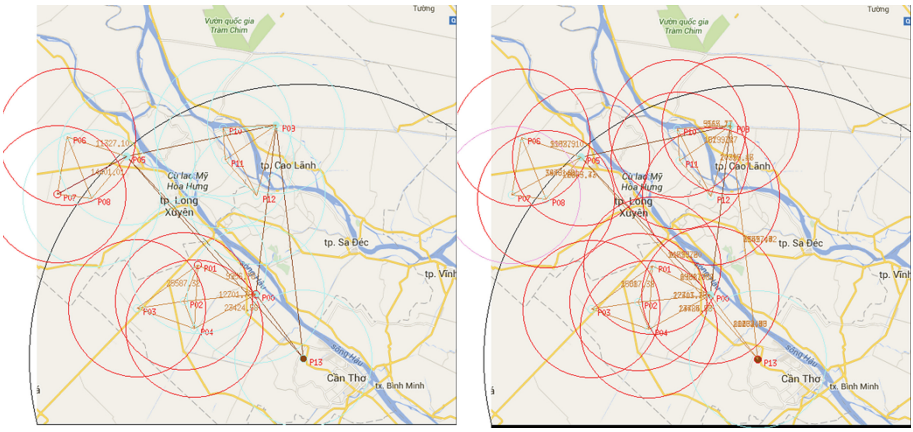

Fig. 11. Sensing and transmitting data in sensor nodes

in their first step to sense and transmit data. In the left, sensor nodes in 3 sub networks of Cantho, Angiang, Dongthap are working to measure environment as well as send data according to routing tables to sink nodes. The sub network at level 0 does not operate yet since it is not the time for its behaviors. On the other hands, the right of the figure illustrates data transmissions in the sub network at level 0. In this case, 3 sub network of Cantho, Angiang, Dongthap do not operate while aggregated data is transmitting through the sub network at level 0 to the gateway.

Figure 12 depicts aggregated data of the BPHSUN after the first 30 min. In the sub network at level 0, because all nodes are in their transmission ranges, aggregated data is transmitted directly to the gateway P13. These pieces of data are mean values of sense data at each sensor node locations after 30 min (2 steps). For example, the row 6 in this figure means that after the first 30 min, the mean value of hoppers collected at P07 (in Angiang sub network) is 9039.72

From	To	Location	Sub network	Mean of hopper density (individuals)
P00	P13	P01	Cantho	8176.4
P00	P13	P02	Cantho	20057.45
P00	P13	P03	Cantho	20302.36
P00	P13	P04	Cantho	19104.9
P05	P13	P06	Angiang	10615.5
P05	P13	P07	Angiang	9039.72
P05	P13	P08	Angiang	21219.6
P09	P13	P10	Dongthap	7341.44
P09	P13	P11	Dongthap	6159.4
P09	P13	P12	Dongthap	15557.32

Fig. 12. Aggregated data in BPHSUN after the first 30 min.

individuals. This mean value is sent from P05 (the sink node of Angiang sub network) to the gateway P13.

6 Conclusion

We have described an hierarchical scheduled algorithm for data dissemination in the BPHSUN, a BPH surveillance network based on WSN approach. In this paper, the BPHSUN forms as an hierarchical structure where light trap sensor nodes place in the same province compose a sub network at level 1 while sink nodes of these sub networks compose another sub network at level 0. In addition, we have depicted a timer to schedule nodes in order to use algorithms for data dissemination to transmit data to the gateway using a routing table.

The hierarchical structure of the model is suitable for management policies of light traps because there are at least 2 management level of traps: province level and region level. It can be considered that sub networks at level 1 is for provincial administrative administration and the sub network at level 0 is for regional management. This structure is simulated with the light trap network in Mekong Delta by using data collections in Cantho, Angiang, Dongthap. The simulation also shows the role of WSN not only in sensing data, but also in collecting data for decision making.

In practice, the distance of light traps is quite far (approximately 10–15 km), LORA technology [17] emerges as a suitable choice for transmitting data to a distant destination. In addition, it is necessary to maintain a database for sensed data as well as aggregated data for post processing. Ongoing investigations include meta data for light trap network to maintain data for post processing.

The paper assumes that data sensed from a sensor node is sent to its neighbors thanks to a routing table. To better routing data, it is necessary to implement some mechanisms such as SYN-ACK or checking/fixing errors during the data transmission.

Acknowledgments. The work of this paper was in the scope of the "OBSNET - Optimizing the Brown Planthopper surveillance network using automatic light traps in the Mekong Delta region" project of the Ministry of Education and Training, Vietnam. The first author was funded by the program 911 of Ministry of Education and Training, Vietnam.

References

1. Conrad, K.F., Fox, R., Woiwod, I.P.: Monitoring biodiversity: measuring long-term changes in insect abundance. In: Insect Conservation Biology, pp. 203–225. Cabi Publisher (2007). ISBN: 9781845932541
2. Crichton, M.I.: The interpretation of light trap catches of Trichoptera from the Rothamsted Insect Survey. In: Malicky, H. (ed.) Proceedings of the First International Symposium on Trichoptera, pp. 147–158. Springer, Netherlands (1976). ISBN: 978-90-6193-547-6

3. http://www.healthyforestpartnership.ca/
4. Rhainds, M., Heard, S.B.: Sampling procedures and adult sex ratios in spruce budworm. Entomol. Exp. Appl. **154**, 91–101 (2014)
5. Rhainds, M.: Wing wear and body size measurements of adult spruce budworms captured at light traps: inference on seasonal patterns related to reproduction. Appl. Entomol. Zool. **50**(4), 477–485 (2015). Springer, Japan. ISBN: 0003-6862
6. Phan, C.H., Huynh, H.X., Drogoul, A.: An agent-based approach to the simulation of Brown Plant Hopper (BPH) invasions in the Mekong Delta. In: 2010 IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), Hanoi, Vietnam, pp. 1–6. IEEE, Heidelberg (2010)
7. Truong, V.X., Huynh, H.X., Le, M.N., Drogoul, A.: Modeling a surveillance network based on unit disk graph technique – application for monitoring the invasion of insects in Mekong Delta region. In: Rahwan, I., Wobcke, W., Sen, S., Sugawara, T. (eds.) PRIMA 2012. LNCS (LNAI), vol. 7455, pp. 228–242. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32729-2_16
8. Lam, H.B., Phan, T.T., Vuong, L.H., Huynh, H.X., Pottier, B.: Designing a brown planthoppers surveillance network based on wireless sensor network approach. In: ISCRAM (Information Systems for Crisis Response and Management) Vietnam 2013 Conference (2013)
9. Lam, B.H., Huynh, H.X., Traoré, M., Lucas, P.Y., Pottier, B.: Monitoring environmental factors in Mekong Delta of Vietnam using wireless sensor network approach. In: 8th International Conference on Simulation and Modelling in the Food and Bio-Industry, FoodSim 2014, pp. 71–78 (2014). ISBN: 978-90-77381-84-7
10. Heong, K.L., Hardy, B.: Planthoppers: new threats to the sustainability of intensive rice production systems in Asia. International Rice Research Institute, Asian Development Bank, Australian Government, Australian Centre for International Agricultural Research (2009). ISBN: 978-90-77381-84-7
11. Lynch, N.A.: Distributed Algorithms. Morgan Kaufmann Publishers Inc., San Francisco (1996). ISBN: 1558603484
12. Pottier, B., Lucas, P.-Y.: Dynamic networks NetGen: objectives, installation, use, and programming. Université de Bretagne Occidentale (2015). https://github.com/NetGenProject
13. NVDIA. https://developer.nvidia.com/cuda-zone
14. WindSonic M. http://gillinstruments.com/data/datasheets/windsonic-m.pdf?v=01.2014
15. DT22. http://www.dfrobot.com
16. BH1750FVI. http://www.electronic-circuits-diagrams.com
17. LORA technology. http://www.semtech.com/wireless-rf/rf-transceivers/sx1276/
18. Taylor, L.R.: Synoptic dynamics, migration and the Rothamsted Insect Survey. J. Anim. Ecol. **55**, 1–38 (1986)
19. Woiwod, I.P., Harrington, R.: Flying in the face of change: the Rothamsted Insect Survey. In: Leigh, R.A., Johnston, A.E. (eds.) Long-term Experiments in Agricultural and Ecological Sciences, pp. 321–342. CAB International, Wallingford (1994)
20. Vutukury, S., Garcia-Luna-Aceves, J.J.: MDVA: a distance-vector multipath routing protocol. In: Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2001, vol. 1, pp. 557–564. IEEE (2001). doi:10.1109/INFCOM.2001.916780
21. Nguyen, K.M., Lam, B.H., Truong, T.P., Thi Pham, H.M., Van Ho, C., Pottier, B., Huynh, H.X.: Automatic hopper light trap. Cantho University Journal of Science, pp. 168–178. Cantho University (2015)