Wireless Sensor Network, 2014, 6, 8-17
Published Online January 2014 (http://www.scirp.org/journal/wsn)
http://dx.doi.org/10.4236/wsn.2014.61002

Scientific Research

Wireless Sensor Network Routing Based on Sensors Grouping

Ammar Hawbani, Xingfu Wang, Yan Xiong, Saleem Karmoshi
University of Science and Technology of China, School of Computer Science and Technology, Hefei, China
Email: ammar12@mail.ustc.edu.cn

Received December 9, 2013; revised December 30, 2013; accepted January 7, 2014

ABSTRACT

Due to the limited communication range of WSN, the sensor is unable to establish direct connection to the data collection station, therefore the collaborative work of nodes is highly necessary. The data routing is one of the most fundamental processes exploring how to transmit data from the sensing field to the data collection station via the least possible number of intermediate nodes. This paper addresses the problem of data routing based on the sensors grouping; it provides a deep insight on how to divide the sensors of a network into separate independent groups, and how to organize these independent groups in order to make them work collaboratively and accomplish the process of data routing within the network.

KEYWORDS

WSN; WSN Routing; Sensors Groups; Groups Routing

1. Introduction

Wireless sensor network consists of a large number of sensors depending on the applications' demands [1-3]. The primary function of these devices is to monitor a natural phenomenon, an environmental phenomenon, or more complicated applications in either medical field or military sphere [1,3]. Sensors are battery-powered devices having a limited lifetime, restricted sensing range, and narrow communication range [4]. The entire shortcomings in sensor networks drive the researchers to develop viable solutions [5,6]. One of the main design aims of WSNs is to transfer data communication while trying to extend the lifetime of the network and avoid connectivity degradation by employing aggressive energy management techniques [7-9].

Due to the limited range of communication, ensuring the direct connection between a sensor and the base station may make the nodes transmit their messages with such a high power that their resources could be quickly depleted. Thus, the collaboration of nodes ensures the communication between distant nodes and base station. In this method, intermediate nodes transmit messages so that a path with multiple links or hops to the base station is established [10-12].

Collaborative work between sensors requires an intelligent organization to transmit information from the sensing field to the base station in order to save energy resources of the network. Because of the insignificant computational capability and the lack of energy sources, the Flooding algorithms are not a proper solution for routing of WSN application [13-15]. The flooding algorithms broadcast the data to all overlapped nodes to the extent that cause an implosion and some nodes redundantly receive multiple copies of the same message. Gossiping algorithm comes with a better performance, avoiding implosion as the sensor and sending the message to a selected neighbor instead of informing all of its neighbors. However, it is still not the proper solution [1,13].

In this paper, we will show an intelligent adaptive solution for data routing so that no node will receive multiple copies of the same message, and there would be no need for Flooding or Gossiping. Our solution here dynamically builds multiple alternative paths. With an adaptive routing, when the sensor has to forward a packet towards the base station, it can choose the path to use

from a set of alternative paths. This selection can be done upon the state of each group's leader (busy or free).

The rest of the paper is structured as follows. Section 2 describes the grouping theory, and a combinatorial computing for intersection areas and intersection points of overlapped sensors. Section 3 describes an algorithm of grouping sensors. In Section 4, we will show an algorithm to manage the wireless sensor network routing. In Section 5, we have shown the evaluation of performance.

2. Sensors Grouping Scheme

Before we starting the routing algorithm we proposed, we would explain what a group of sensors is.

Two sensors creates three regions if they overlapped see **Figure 1**. The function $\Delta(s_1, s_2) = \{s_1 - s_2, s_2 - s_1, s_1 \cap s_2\}$ is expressing the created regions. For convenient, we will express the created regions by $\Delta(s_1, s_2) = \{R_1, R_2, R_3\}$. The symbol $|\Delta|$ denotes to the number of created regions. For example $|\Delta(s_1, s_2)| = 3$. Each region of $\{R_1, R_2, R_3\}$ has a coverage degree. The symbol $\beta(R_i)$ denotes the coverage degree.

$$\beta(R_1) = 1, \beta(R_2) = 1, \beta(R_3) = 2.$$

The symbol $\omega(R_i) = \{s_1, s_2, \cdots\}$ denotes the set of sensors covered the R_i region.

$$\omega(R_1) = \{s_1\}, \omega(R_2) = \{s_2\}, \omega(R_3) = \{s_1, s_2\}.$$

A group of k sensors $G_k = \max\{\omega(R_1), \omega(R_2), \omega(R_3), \cdots, \omega(R_{1+k(k-1)})\}$. For example,

$$G_1 = \max\{\omega(R_1), \omega(R_2), \omega(R_3)\} = \omega(R_3) = G_2 = \{s_1, s_2\}.$$

In [16], the authors provided an analysis about grouping strategy of WSN. We will used this idea to divide the WSN into groups and then organize these groups in order to facilitate the routing process. This grouping scheme is not useful if the sensors moving quickly.

3. Sensors Grouping Algorithm

In this section, we will provide recursive algorithm to divide the overloaded sensors in to separated groups. A deep explanation is provided in **Appendix**. We will start by define the terms to be used in our algorithm.

3.1. Definitions

Group of **Sensors** G_i: is a set of sensors that are overlapped while covering the same region in the sensing field.

$$G_i = \{s_1, s_2, s_3, \cdots, s_j \mid \exists p(x, y) \in s_1, s_2, s_3, \cdots \text{ and } s_j\}$$

In the this paper, we will use the notation $s_i \cap s_j$ to indicate the sensor s_i overlapped with sensor s_j. More precisely, s_i and s_j covered the same region. The group of sensors is an object containing the members shown below.

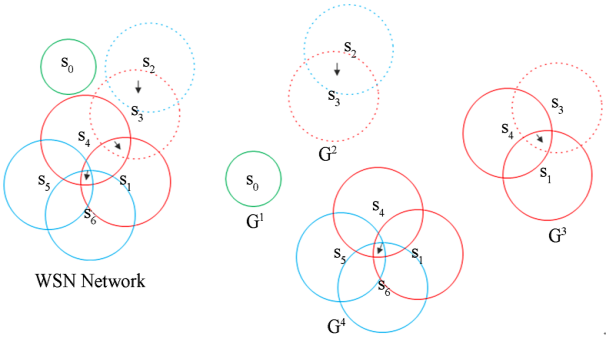


Figure 1. Partition the WSN into sensors' group.

```
Class SensorGroup
    {
intGroupID { get; set; }
intGroupLength { get; set; }
stringGroupMembersString { get; set; }
stringGroupingMethod { get; set; }
List<Sensor>GroupMember{ get; set;}
    }
```

Vector of **a Sensor** V_i: is a set of sensors, which are overlapped with the sensor s_i.

$$V_i = \{s_1, s_2, s_3, \cdots, s_j \mid s_i \cap s_1, s_i \cap s_2, s_i \cap s_3, \cdots \text{ and } s_i \cap s_j\}$$

Each sensor in the network has a vector. Some of the sensors inside the network have the same vector. The vector is an object:

```
class Vector
    {
intSensorID { get; set; }
int Length { get; set; }
stringSensorsOverlappingString { get; set; }
        List<Sensor> Sensors { get; set; }
        Ellipse SensorBody{ get; set; }
        Sensor Sensor{ get; set; }

    }
```

Direct Group $[G_i]$**:** is a group of sensors that fully matches a vector in the network.

Indirect Group $\langle G_i \rangle$**:** is a Hybrid and collected group of sensors by processing multiple vectors.

Matched Direct Groups of a Vector: The sensors in ungrouped vectors $\|\langle V \rangle\|$ are not fully matched with any direct group $\|[G]\|$, but they can matched some sensors in multiple direct groups, some sensors of ungrouped vectors might not be matched with any sensors in the directed groups.

Suppose an ungrouped vector:

$$\langle V_k \rangle = \{s_1, s_2, s_3, \cdots, s_k\},$$

A list of direct groups:

$$\|[G]\| = \{[G_1],[G_2],[G_3],\cdots,[G_i]\},$$

Each group contain a number of sensors

$$[G_j] = \{s_1, s_2, s_3, \cdots, s_j\}.$$

The matched group:

$$M_i = [G_j] \subsetneq V_k$$

Remnant Sensors: remnant sensors R_i are those sensors in ungrouped vectors $\langle V_i \rangle$ that do not matches any sensor in direct group.

$$R_i = \langle V_i \rangle - M_i$$

Solid Vector: The solid vector $\bar{S}$ is the union of remnant sensors $\|R\|$ of each ungrouped vectors.

$$\bar{S} = \bigcup_{R_i \in \|R\|} R_i$$

Filtered Vectors: By finding the solid vector $\bar{S}$, we can exclude all the sensors directly grouped already and also find the sensors that are not grouped yet. For each ungrouped vector $\langle V_i \rangle$,

$$F_i = V_i \cap \bar{S}$$

3.2. System Model

The wireless sensor network is a list of vectors collected by all sensors in the field. The network of n sensors is represented by a square matrix $(n \times n)$.

$$A^* = \begin{bmatrix} 1 & 1 & & 0 \\ 1 & 1 & & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 1 & & 1 \\ 0 & 0 & & 1 \end{bmatrix}_{n \times n}$$

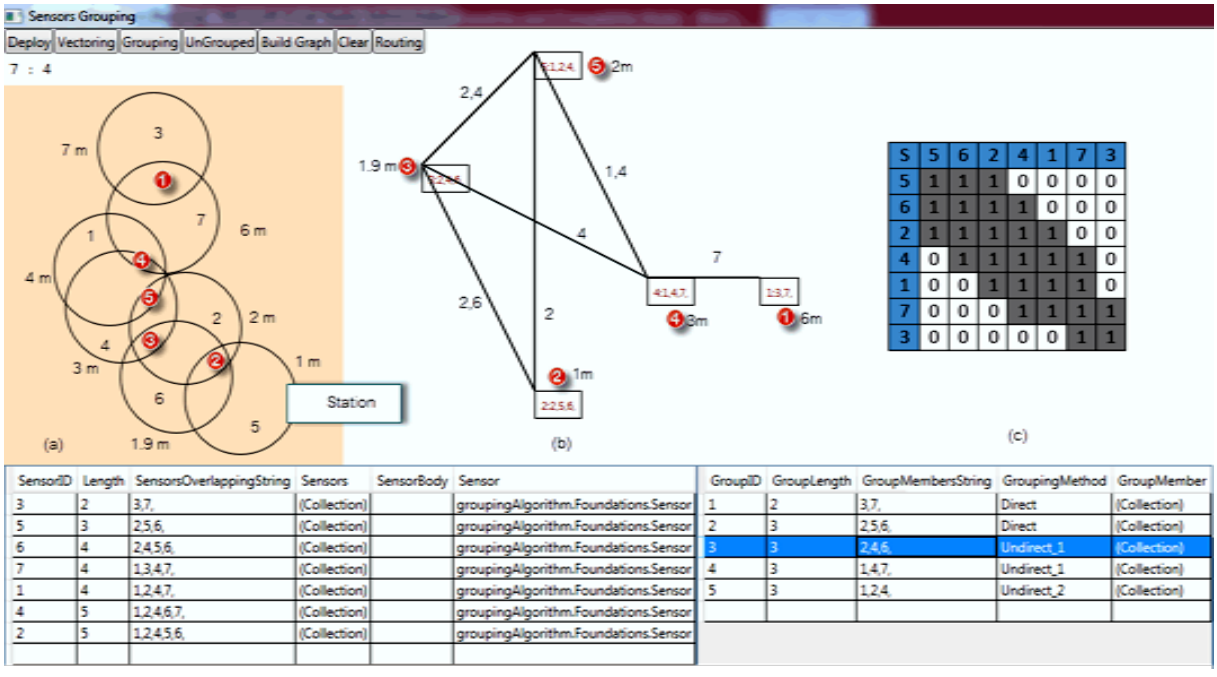
To facilitate the calculations, assume that the numerical value of sensors in the matrix is either 1 or 0 as below.

$$a_{ij} = \begin{cases} 1 & s_i \text{ and } s_j \text{ are overlapped} \\ 1 & i = j \\ 0 & s_i \text{ and } s_j \text{ are not overlapped} \end{cases}$$

We can find the maximum coverage degree areas by partitioning A^* into square sub-matrices $\{A_1, A_2, A_3, \cdots\}$, such as the value of each sensor of sub-matrix $a_{ij} = 1$ as well as the sub-matrix contain the maximum possible number of rows and columns. Each sub-matrix should match a group of sensors. For example, the network in **Figure 2** is partitioned into five square sub-matrices shown as below.

$$A^* = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}_{g1=(3,7)} \cup \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{g2=(2,5,6)} \cup \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{g3=(2,4,6)}$$

$$\cup \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{g4=(1,4,7)} \cup \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{g5=(1,2,4)}$$



S	5	6	2	4	1	7	3
5	1	1	1	0	0	0	0
6	1	1	1	1	0	0	0
2	1	1	1	1	1	0	0
4	0	1	1	1	1	1	0
1	0	0	1	1	1	1	0
7	0	0	0	1	1	1	1
3	0	0	0	0	0	1	1

SensorID	Length	SensorsOverlappingString	Sensors	SensorBody	Sensor
3	2	3,7,	(Collection)		groupingAlgorithm.Foundations.Sensor
5	3	2,5,6,	(Collection)		groupingAlgorithm.Foundations.Sensor
6	4	2,4,5,6,	(Collection)		groupingAlgorithm.Foundations.Sensor
7	4	1,3,4,7,	(Collection)		groupingAlgorithm.Foundations.Sensor
1	4	1,2,4,7,	(Collection)		groupingAlgorithm.Foundations.Sensor
4	5	1,2,4,6,7,	(Collection)		groupingAlgorithm.Foundations.Sensor
2	5	1,2,4,5,6,	(Collection)		groupingAlgorithm.Foundations.Sensor

GroupID	GroupLength	GroupMembersString	GroupingMethod	GroupMember
1	2	3,7,	Direct	(Collection)
2	3	2,5,6,	Direct	(Collection)
3	3	2,4,6,	Undirect_1	(Collection)
4	3	1,4,7,	Undirect_1	(Collection)
5	3	1,2,4,	Undirect_2	(Collection)

Figure 2. Implementation of the grouping algorithm graph. (a) Sensing field; (b) grouping graph; (c) grouping matrix.

We can see that each group of overlapped sensors is a square matrix, and the value of each element of the matrices is 1. Simply, we just find out the square matrices with the following conditions: The maximum possible number of rows and columns should be alighted together and the value of each element is 1. Suppose that each sensor s_i in the field sends a vector $v_i = [s_i \quad s_1 \quad s_2 \quad \cdots \quad s_m]$ to the base station, all vectors create the A^*. Let $k = |v_i|$ be the number of sensors of v_i. In addition to that, let R be the number of repetition of v_i in A^*. The first step of this algorithm is to find the sensors overlapping relations based on their distance; each sensor has a vector of sensors. The second step is to sort these vectors according to the number of sensors within.

3.3. Groups Algorithm

Finding a group of sensors indirectly can be manipulated by dividing ungrouped vectors into smaller vectors so that the sensors that are directly grouped already are not needed. To do so, we can follow the steps below:

1) Find ungrouped vectors by comparing the direct groups and the network vectors list (GetUnGroupedVectors $(\|V\|, \|[G]\|)$).

2) Find the matching direct groups associated to ungrouped vectors (GetMatchingGroupsForVector $(\|[G]\|, \|\langle V\rangle\|)$).

3) Find the solid vector, which contains the remnant sensors of ungrouped vectors list (GetSolidVector $(\|M\|)$).

4) Filter ungrouped vectors according to the solid vector (FilterUnGroupedVectorsAccordingtoSolidVector $(\|\langle V\rangle\|, \bar{S})$).

5) Extract new groups from filtered ungrouped vectors $\|F\|$ by calling the method of finding the direct groups (GetDirectGroups $(\|F\|)$).

6) Repeat the steps from step 1 to step 5 recursively until no new groups are found (**Algorithm 1**).

Recursive algorithms solve the problem by solving smaller versions of the same. The smaller versions of ungrouped vectors are about half the size of the original vectors. The algorithm can be referred to as a "divide and conquer" algorithm. Say that we have (*n*) of original vectors, the time needed to extract (*g*) direct groups is $g(n)$. However, not all vectors are able to be directly grouped. For ungrouped vectors, there will be (n-g) vectors. The recursive time needed is:

$$T(n) = g(n) + T(n-g)$$

$$T(n) = \begin{cases} g(n) \text{ directgroups} \\ T(n-g) \text{ indirectgroups} \end{cases}$$

ALGORITHM 1: GetInDirectGroups
FINDING THE DIRECT GROUPS.

INPUT: $\|V\|, \|[G]\|$

OUTPUT: $\|\langle G\rangle\|$

```
1)  ||⟨V⟩|| = Un Grouped Vectors(||V||, [G])
2)  if (V != null)
3)   {
4)  ||M|| = Get Matching Groups For Vector(||[G]||, ||⟨V⟩||)
5)  S̄ = get Solid Vector(||M||)
6)  ||F|| = Filter Un Grouped Vectors According to Solid Vector(||⟨V⟩||, S̄)
7)  [G_i] = Get Direct Groups (||F||)
8)  if ([G_i]! = null)
9)      {
10) ||[G]||.add([G_i])
11) Get In Direct Groups(||⟨V⟩||, [G_i])
12)                                                    }
13)                                                    }
```

Algorithm 1. Finding the direct groups.

The analysis depends on the preparation work to divide the input, the size of the ungrouped vectors, the number of recursive calls and the concluding work to combine the results of the recursive calls.

$$T(n) = n + T(n-g)$$

g is the number of extracted groups of each recursive call.

4. Routing Basing on Group of Sensors

4.1. Sensors Graph

The routing algorithm is started by dividing the wireless sensors network to independent groups (explained in Subsection 3.3). Each group is comprised of a certain number of sensors. A Sensor may belong to more than one group. If a sensor belongs to group A and belongs to group B, we say there is a link between group A and B, and we call this sensor by coordinator or the group leader. Each group contains one or more leaders.

The graph of sensor network $G = (V, E)$ consists of a finite nonempty set *V* of groups called vertices and a set E of 2-elements of *V* called edges. In the graph theory, the notation *V*(*G*) is the set of vertex and *E*(*G*) is a set of edges. Using the idea of graph theory, we can say that each vertex is represented by a group of sensors. The leader of the group is represented with an edge. As long

as the graph is connected, there will be a path from the source node connecting the base station, hence the packet must be reaching the sink node. Here we assume that connectivity and coverage of network are managed well.

The graph in **Figure 2** can be represented mathematically by

$$V(G) = G^* = \{g_1, g_2, g_3, g_4, g_5\},$$

$$E(G) = \{(g_1, g_4)_7, (g_4, g_5)_1, (g_4, g_5)_4, (g_4, g_3)_4,$$
$$(g_5, g_3)_2, (g_5, g_3)_4, (g_3, g_2)_2, (g_3, g_2)_6, (g_2, g_5)_2\}.$$

The degree $d(v)$ of a vertex V is its number of incident edges. Any two groups have an incident edges called to be neighbor groups. Each group has at least one neighbor otherwise the network is disconnected. Let $N(g) = \{g_1, \cdots\}$ is the neighbor groups of g for instance in **Figure 2**, $N(g_5) = \{g_2, g_3, g_4, g_6\}$.

The distance between the source node and the sink node is an important parameter to control and improve the performance of packet forwarding in the overall network. Considering the power consumption, the nearest nodes to the sink could save more power by building a shorter path with a minimum number of hops. Since our algorithm is based on grouping, we need to define the term of grouping distance, the distance of group is the least distance among sensors inside the group to the base station. Should we denote to the distance between sensor s1 and the base station by $D(s_1)$. In **Figure 2**,

$$D(g_5) = \min\{D(S_1), D(S_4), D(S_8)\}.$$

We assumed each that group is taking into account the information of its neighbor groups' distance. With adaptive routing, when a source group has to forward a packet towards a particular group, it can choose the leader sensor to use from a set of alternative leaders associated to the group. This selection can be done upon two conditions: the first condition is the current state of the leader (busy or free) and therefore, the busy leaders are skipped. The second condition is the least distance to the base station and therefore, the nearest group to base station can be selected.

4.2. Finding Leaders

The Leader sensor acts as direct link among its associated groups. It can keep forwarding the data packet to all groups it belongs to. Straightforwardly, we can list the leaders by the simple algorithm below (**Algorithm 2**).

4.3. Selecting the Leader

Each group has a set of neighbors and a set of leaders. When a packet has been forwarded to a group, the group should know well its leaders and neighbors and therefore make the decision of packet routing to the next hop. In the source group, there are one or more leaders connecting to one or more neighbors.

```
Input: K of sensors
Output: Leaders
Find   G*
For each  (G^i ∈ G*)
{
    For each  (G^{i+1} ∈ G*)
    {
        If  (G^i ∩ G^{i+1} ≠ ∅)
        {
E.add(G^i, G^{i+1}) ;
        }
    }
}
```

Algorithm 2. Finding the leaders.

Multiple leaders in the source node might link a single neighbor. In addition, the leader might connect to multiple neighbors. Thus, after selecting the nearest neighbor group, we must ensure that the connecting leader is free, otherwise, the packet cannot be forwarded via this leader. If the leader is busy, then the packet must be forwarded to the second nearest neighbor. If there is only one leader in the source node, the packet should be delayed until the leader becomes free.

The selection process of the leader is running simply by choosing the nearest neighbor and checking the availability of the leader connected to this neighbor. If the leader is free, then this is the adaptive channel to forward the data. There will be more than one adaptive channel when there are more than one leaders (parallel leaders) connected to nearest neighbor. If the current state of all parallel leaders is busy, then there will be two ways to deal with: the first way, the packet should wait until one of the parallel leaders becomes free. However, this is not respectable in case the application's demand is a real time stream of monitoring. The second way: if the source node contains other leaders connecting to other neighbors, the packet can be forwarded to any other neighbor. However, this might lead to an increment of the number of routing hops, hence might maximize the usage of energy in the overall network, this might lead to the death of a sensor. If there is only one neighbor associated to the group and all leaders are busy, then the first way is obligatory. In case of all parallel leaders are free, any leader election algorithm can be applied to manage the selection of the leader.

Leaders in the same group are called partners. Moreover, if more than one leader is connected to the same neighbor group, they are called twin leaders. A set of leaders in a group can be partners or twins, neighbors can determine this. For example, in **Figure 2**, group 5 contains three elements, three of them are leaders, and thus

leaders $\{(g_4,g_5)_1,(g_4,g_5)_4\},\{(g_5,g_3)_2,(g_5,g_3)_4\}$ are twins. If a leader connected to more than one neighbor-group is called identical leader, for example, $(g_4,g_3)_4$ and $(g_4,g_5)_4$ are identical Leaders.

In **Method 1**, the input is the source node (SensorGroupSourceGroup), and a list of neighbors associated to the source node (List<Neighborgroups>Neighbors), on the other hand, the output is the selected leader. This way, first, the nearest neighbor is selected, and then we should find the leaders of the source node, which are connected to the selected neighbor. The forwarding decision of packets is deterministic and adaptive in each source group (**Algorithm 3**).

4.4. Numerical Example

As shown in **Figure 2**, (a) sensors are deployed in the field. Say, sensor (3) detects a target. Here we assume that all leaders are free. The packet routing from sensor (3) to the station is going according to the steps below (see **Figure 3**). The source groups (G^1) have one neighbor and one leader, forwarding data towards group (G^4) via sensor (7) obligatory. When the packets arrived to (G^4), it has four leaders and three neighbors, the min group distance is to (G^3), hence the next hop is (G^3) via sensor (4). After the packet has arrived to (G^3), this group (5) leaders and three neighbors, the min group distance is to (G^2), Thus, the next hop is (G^2) via sensor (2).

```
Input: a group of sensors, and a list of Neighbor groups.
Output: a sensor called leader.
Sensor SelectLeaderSensor(SensorGroup
SourceGroup ,List<Neighborgroups> Neighbors)
       {
           Sensor Leader = null;
           Neighborgroups SelectedNeighbor = SelectMinDis-
tanceNeighbor(Neighbors);
           List<Sensor> LeadersAssociatedwithSelected-
Neighbor =
           LeaderAssociatedwithNeighbor(SourceGroup, Se-
lectedNeighbor);
           Sensor selectFreeSensor =
           SelectFreeSen-
sor(LeadersAssociatedwithSelectedNeighbor);
           Leader = selectFreeSensor;
           return Leader;
       }
```

Method 1. **Select leader.**

Input: g of groups
Output: Forward packet
Find G^*
For each $(G^i \in G^*)$
{
  For each $(G^j \in N(G^i))$
  {
Lsensor=SelectLeaderSensor (G^i,G^j);
Forward(Date,*Lsensor*);
  }
}

Algorithm 3. **Routing basing on groups.**

Source node (G1)		
Neighbor	D	Leader
G4	3	7
Next hop (G4)		
Selected Leader (7)		

(a)

Source node (G4)		
Neighbor	D	Leaders
G1	3	7
G3	1.9	4
G5	2	1,4
Next hop (G3)		
Selected Leader (4)		

(b)

Source node (G3)		
Neighbor	D	Leaders
G2	1	2,6
G4	3	4
G5	2	2,4
Next hop (G2)		
Selected Leader (2)		

(c)

Figure 3. **Routing tables in each source node from (G1 to G2) when all sensors are free.**

5. Performance Evaluation

In this section, we will evaluate the impact on network performance of the proposed routing algorithm. For this purpose, we have developed a detailed simulator that allows us to estimate the network performance, power consumption, and the number of hops. The results are shown in the **Figures 4-6**.

Counting the Average Number of Hops

The number of hops depends on the number of groups. Say we have n nodes deployed randomly in the sensing field, and want to compute the number of possible groups that can be generated. A pattern of groups is a deployment way for sensors groups such that all sensors in the group are connected. Let's start by a simple example with n = 4. As shown in **Table 1** and **Figure 7**, there are four grouping patterns.

Let us denote to the pattern by p_i and to the number of hops of each pattern by h_i.

The expression of patterns can be written as:

$$E(n)=\sum_{i\in p_i} E_i$$

Easily we can see the patterns of four sensors are:

$$E(4)=G^3\cdot G^2+3G^2+2G^3+G^4$$

For five sensors, there will be eight patterns.

$$E(5)=G^4\cdot G^3+G^4\cdot G^2+2G^3\cdot G^2+G^3\cdot 2G^2$$
$$+4G^2+3G^3+2G^3+G^5$$

Let us donate to the number of pattern by $P(n)$. For n sensors, there will be $P(n)=2^{n-2}$ patterns.

The number of hops is changed according to patterns. Let $h(n)$ be the sum of hops of all patterns. For example, $h(4)=3+2+2+1=8$ as shown in **Table 1** and **Figure 7**, it is easy to find thatthe average number of hops is

$$h(n)=\sum_{i\in E(n)} h_i=\frac{n}{2}2^{n-2}$$

Table 1. Grouping patterns of four sensors deployed (see Figure 7).

Pattern p_i	Expression E_i	Number of hops h_i
A	$3G^2$	3
B	$G^2 \cdot G^3$	2
C	$2G^3$	2
D	G^4	1

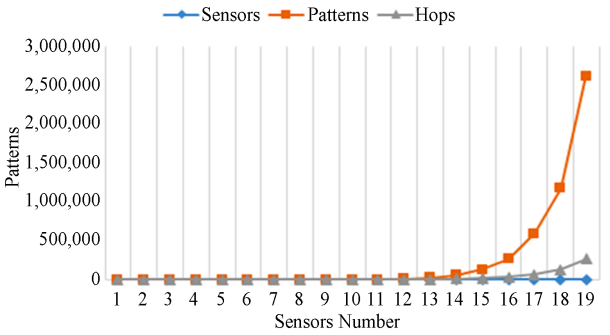


Figure 4. Pattern count and hops.

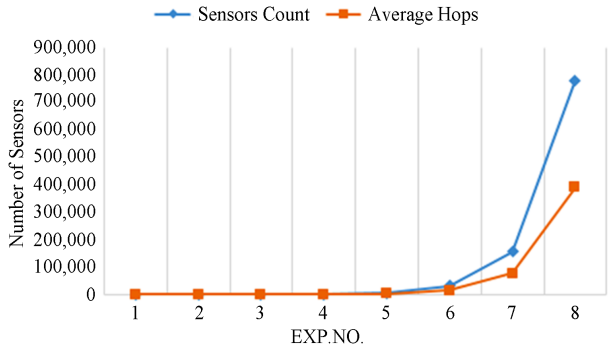


Figure 5. Evaluation of average routing performance based on the grouping algorithm.

$$Av(n) = \frac{h(n)}{E(n)} = \frac{\frac{n}{2}2^{n-2}}{2^{n-2}} = \frac{n}{2}$$

In the second experiment, there are 10,000 nodes deployed in different coverage degree. Say we have N sensors deployed randomly. Let us say that the degree of network coverage is C, the complexity of algorithm (N/C) (See Figure 6).

6. Conclusion

We have proposed an algorithm where the source group forwards a packet to one neighbor only and there is no need of flooding or forwarding packets to all neighbors. The grouping adaptive routing saves more power therefore, ends up maximizing the lifetime of the wireless sensor network.

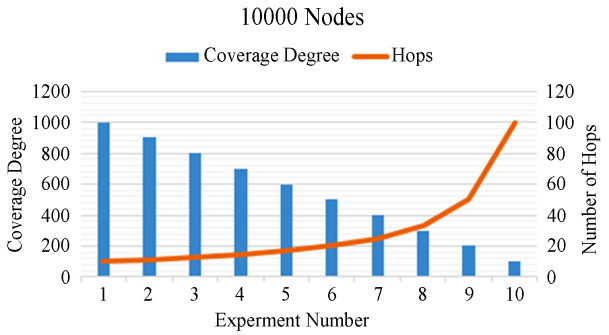


Figure 6. Evaluation of routing performance based on the grouping algorithm on different coverage degrees.

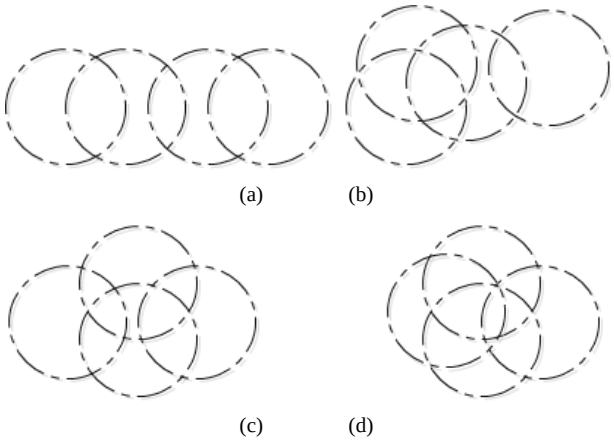


Figure 7. Grouping patterns of four sensors deployed.

Acknowledgements

The authors would like to acknowledge The National Natural Science Foundation of China, the National Science Technology Major Project and the China Scholarship Council for their supports.

REFERENCES

[1] L. J. G. Villalba, A. L. S. Orozco, A. T. Cabrera and C. J. B. Abbas, "Routing Protocols in Wireless Sensor Networks," *Sensors*, Vol. 9, No. 11, 2009, pp. 8399-8421.

[2] E. Zanaj, M. Baldi and F. Chiaraluce, "Efficiency of the Gossip Algorithm for Wireless Sensor Networks," In *Proceedings of the* 15*th International Conference on Software, Telecommunications and Computer Networks* (*SoftCOM*), Split-Dubrovnik, September 2007.

[3] J. C. Martinez, J. Flich, A. Robels, P. Lopez and J. Duato, "Supporting Adaptive Routing in infiniband Networks," *Proceedings of the* 11*th Eromicro Conference on Parallel, Distributed and Network-Based Processing* (*Euro-pdp*03) 2003.

[4] S. C.-H. Huang, S. Y. Chang, H.-C. Wu and P.-J. Wan, "Analysis and Design of Novel Randomized Broadcast Algorithm for Scalable Wireless Networks in the Interference Channels," *IEEE Transactions on Wireless Communications*, Vol. 9, No. 7, 2010, pp. 2206-2215.

[5] H. H. Zhang and J. C. Hou, "Maximising α-Lifetime for Wireless Sensor Networks," 2007.

[6] M. Cardei and D. Z. Du, "Improving Wireless Sensor Network Lifetime through Power Aware Organization," *Wireless Networks*, Vol. 11, No. 3, 2005, pp. 333-340.

[7] S. Poduri and G. Sukhatme, "Constrained Coverage for Mobile Sensor Networks," *IEEE International Conference on Robotics and Automation*, New Orleans, April 26-May 1, 2004, pp. 165-172.

[8] A. Chen, S. Kumar, and T.-H. Lai, "Designing Localized Algorithms for BarrierCoverage," MOBICOM. ACM, 2007.

[9] X. Bai, Z. Yun, D. Xuan, T. Lai and W. Jia, "Optimal Patterns for Four-Connectivity and Full Coverage in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, 2008.

[10] J. Wu and S. Yang, "Coverage and Connectivity in Sensor Networks with Adjustable Ranges," *International Workshop on Mobile and Wireless Networking* (*MWN*), 2004.

[11] M. Cardei, J. Wu, N. Lu and M. O. Pervaiz, "Maximum Network Lifetime with Adjustable Range," *IEEE Intl. Conf. on Wireless and Mobile Computing, Networking and Communications* (*WiMob*'05), 2005.

[12] M. Cardei, M. T. Thai, Y. Li and W. Wu, "Energy-Efficient Target Coverage in Wireless Sensor Networks," *Proceedings of IEEE Infocom*, 2005.

[13] M. Cardei and D.-Z. Du, "Improving Wireless Sensor Network Lifetime through Power Aware Organization," *Wireless Networks*, Vol. 11, No. 3, 2005, pp. 333-340.

[14] J. K. Hart and K. Martinez, "Environmental Sensor Networks: A Revolution in the Earth System Science?" *Earth-Science Reviews*, Vol. 78, No. 3, 2006, pp. 177-191.

[15] Y. Ma, M. Richards, M. Ghanem and Y. Guo, "Air Pollution Monitoring and Mining Based on Hassard Sensor Grid in London," *Sensors*, Vol. 8, No. 6, 2008, p. 3601.

[16] H. Ammar, X. F. Wang and Y. Xiong "Sensors Grouping Model for Wireless Sensor Network," *Journal of Sensor Technology*, Vol. 3, No. 4, 2013, pp. 133-140.

Appendix

Listing of Vectors

According to **Method 1**, **GetVectorsList**, the input $(\|S\|)$ is a list of sensors deployed in the sensing field. The output $(\|V\|)$ is a list of vectors. The main procedure here is to find the overlapped sensors with the required sensor $s_i\,(line\ 6)$. Each sensor s_i finds the overlapping list of sensors and sends it to the base station where the square matrix of network A^* is built. The algorithm for finding the groups will run in the base station. In case the grouping process run internally in the local node, each node send its vector to the adjacent node only. Here we suppose the grouping process will run externally.

METHOD 1: GetVectorsList
FINDING THE VECTORS FOR EACH SENSOR IN THE NETWORK
INPUT: $\|S\|$
OUTPUT: $\|V\|$

1) $\forall s_i \in S$
2) *do*
3) {
4) $V_i \cdot id = i$
5) $V_i \cdot sensor = s_i$
6) $V_i = Find\_Over(s_i, \|S\|)$
7) $V \cdot add(V_i)$
8) }

Direct Grouping

In the **Method 2**, **GetDirectGroups,** the input of algorithm $(\|V\|)$ is a list of vectors of all sensors deployed in the field. The output $(\|[G]\|)$ is a list of direct groups extracted from the list of vectors.

METHOD 2: GetDirectGroups
LIST OF DIRECT GROUPS
INPUT: $\|V\|$
OUTPUT: $\|[G]\|$

1) $\forall V_i \in \|V\|$
2) *do*
3) {
4) Integer k = Repetition_of_Vector$(V_i, \|V\|)$
5) If $(k == |V_i|)$
6) {
7) $[G_i] = V_i$
8) $\|[G]\| \cdot \mathbf{add}([G_i])$
9) }
10) }

Ungrouped Vectors

For each vector V_i in the network list $\|V\|$, if not matched any entry $[G_i]$ in the direct groups list $\|[G]\|$, then it considered as ungrouped vector $\langle V_i \rangle$.

$$\|\langle V \rangle\| = \{V_1, V_2 \cdots V_j \mid V_i \in \|V\| \text{ and } \notin \|[G]\| 1 \le i \le j\}.$$

In the **Method 3**, **GetUnGroupedVectors,** the inputs are $(\|V\|)$ a list of the network vectors and the direct groups $(\|[G]\|)$. Therefore, we just find those vectors, which are not grouped yet.

METHOD 3: GetUnGroupedVectors
LIST THE INDIRECT VECTORS.
INPUT: $\|V\|, \|[G]\|$
OUTPUT: $\|\langle V \rangle\|$

1) $\forall V_i \in \|V\|$
2) *do*
3) {
4) If $(!\text{IsVectorGroupedAlready}(\|[G]\|, V_i))$
5) {
6) $\|\langle V \rangle\| \cdot add(V_i)$
7) }
8) }

Finding the Matching Direct Groups of a Vector

In the **Method 4**, **GetMatchingGroupsForVector,** the input $(\|[G]\|)$ is a list of direct groups and $(\|\langle V \rangle\|)$ is a list of ungrouped vectors. The output $(\|M\|)$ is a list of the direct groups which matched the ungrouped vectors. For each ungrouped vector, we will find the matching direct groups by dividing the ungrouped vectors into smaller vectors. Each ungrouped vector might match multiple direct groups.

METHOD 4: GetMatchingGroupsForVector
FINDING THE DIRECT GROUPS WHICH MATCH UNGROUPED VECTORS
INPUT: $\|\langle V \rangle\|, \|[G]\|$
OUTPUT: $\|M\|$

1) $\forall V_j \in \|\langle V \rangle\|$
2) *do*
3) {
4) $\forall [G_i] \in \|[G]\|$
5) *do*
6) {
7) $M_j = [G_i] \subsetneq \langle V_j \rangle$
8) $If\ (M_j != \varnothing)$
9) $M \cdot add(M_j)$
10) }
11) }

Remnant Sensors

In the **Method 5**, **RemnantSensors,** the inputs are $(\|M\|)$, a list of matched groups, and $(\langle V_i \rangle)$, ungrouped vector. The output is a list of remnant sensors $\|R\|$. The first step is to find the union of matching direct groups associated with the ungrouped vector, and to list them in (unionMembersOfGroup), then find the interaction of (unionMembersOfGroup) with the vector's sensors.

METHOD 5: RemnantSensors
FINDING THE REMNANT SENSORS
INPUT: $\langle V_i \rangle \|M\|$
OUTPUT: R_i

1) $\forall V_i \in \|\langle V \rangle\|$
2) *do*
3) {
4) $\forall M_k \in \|M\|$
5) *do*
6) {
7) $R_i = \langle V \rangle_i - M_k$
8) $If\ (R_i != \varnothing)$
9) $\|R\| \cdot add\,(R_i)$
10) }
11) }

Solid Vector

Most of the sensors in the solid vector do not exist in the direct groups. This vector runs as a filter for ungrouped vector, and only those sensors which appear in the solid vector can appear in the ungrouped vector as well. See **Method 6**.

METHOD 6: getSolidVector
THE LIST OF REMNANT SENSORS
INPUT: $\|R\|$
OUTPUT: $\bar{S}$

1) $\bar{S}$
2) $\forall R_i \in \|R\|$
3) *do*
4) {
5) $\bar{S} \cup = R_i$
6) }

Filtered Vectors

After filtering all ungrouped vectors, we can continue counting the repetition of all filtered ungrouped vectors until we find new direct groups. In **Method 7**, the inputs are $(\bar{S})$, a solid vectors, and $\|\langle V \rangle\|$, a list of ungrouped vectors. The output is a list of vectors contains only those sensors that appeared in the solid vector.

METHOD 7:FilterUnGroupedVectorsAccordingtoSolidVecto
FINDING THE FILTERED VECTORS
INPUT: $\|\langle V \rangle\|, \bar{S}$
OUTPUT: $\|F\|$

1) $\forall V_i \in \|\langle V \rangle\|$
2) do
3) {
4) $F_i = \langle V_i \rangle \cap \bar{S}$
5) $\|\mathrm{F}\| \cdot add\,(F_i)$
6) }