International Journal of Innovative Computing, Information and Control
Volume **9**, Number **10**, October **2013**

ICIC International ©2013 ISSN 1349-4198
pp. 4085–4099

DYNAMIC CLUSTER-BASED ARCHITECTURE AND DATA CONGREGATION PROTOCOLS FOR WIRELESS SENSOR NETWORK

A. K. M. Muzahidul Islam[1,*], Koichi Wada[2] and Wei Chen[3]

[1]Malaysia-Japan International Institute of Technology
University Teknologi Malaysia
54100 Jalan Semarak, Kuala Lumpur, Malaysia
[*]Corresponding author: akmmislam@ic.utm.my

[2]Department of Applied Informatics
Hosei University
3-7-2 Kajino-cho, Koganei 184-8584, Japan
wada@hosei.ac.jp

[3]Department of Computer Science
Tennessee State University
3500 Johns A Merrit Blvd, Nashville, TN 37209, USA
wchen@tnstate.edu

Received September 2012; revised January 2013

Abstract. *In recent years, wireless sensor networks have changed the way we have started looking into the world. It has shown its potential ability in providing solutions in various areas such as health care, environment, defense, surveillance, industry and transport. Typically, the sensor nodes of a Wireless Sensor Network have a large coverage area and longer range. Moreover, they are self-configuring or self-organizing. Clustering provides hierarchical organization to a flat sensor network topology. Sometimes, a Wireless Sensor Network is called as dynamic if it is supported by two operations, namely node-move-in and node-move-out. Node-move-in refers to nodes' joining into an existing network, whereas node-move-out denotes nodes' getting out of a network. Data congregation is one of the fundamental network operations on Wireless Sensor Network where data is collected from the network in some nodes and once collected the data is then forwarded to some sink nodes in order to perform some specific task. In this paper, we propose time-efficient data congregation protocols for Dynamic Cluster-based Wireless Sensor Network (CBWSN). First, a dynamic cluster-based architecture is presented. Then proposed data congregation protocols are exhibited. Primarily, it is shown that data congregation can be done in* $O(p+\Delta)$ *intervals, i.e., time-slots, where* p *is the number of clusters in the network;* Δ *is the maximum degree of nodes in the cluster-based architecture. The cluster-based architecture is further improved to facilitate a time-efficient flooding protocol and a better data congregation technique. In this work, flooding and data congregation are done in* $O(h)$ *and* $O(h+\Delta)$ *time-slots, respectively, where* h *is the height of the cluster-based architecture. Finally, some simulation results have been exhibited to show the efficiency of the proposed architecture and protocols.*

Keywords: Dynamic, Cluster-based wireless sensor network, Node-move-in, Node-move-out, Broadcasting, Gathering

1. **Introduction.** Typically, Wireless Sensor Networks (WSNs) contain hundreds or thousands of sensor nodes that have the ability to communicate either among each other or directly with an external base station. In recent years, WSNs, the dense wireless networks of sensor nodes collecting and disseminating environmental data, have received a tremendous amount of attention in research. There are various scenarios in which such networks might find applications, such as environmental control in office buildings, robot

control and guidance in automatic manufacturing environments, smart home [1,24,25]. The sensor networks have a large coverage area and longer range than other wireless networks. They have a higher degree of fault tolerance than other wireless networks: failure of one or a few nodes does not affect the operation of the network. They are also self-configuring and self-organizing [2].

In this paper, we focus on data congregation (also called as data gathering) problem on a large-scale wireless sensor network to apply in a variety of applications in infrastructure and habitat monitoring.

In common, data transmissions are accomplished through multi-hop routing from individual sensor nodes to a data sink. Successful deployment of such networks faces the challenge in effective global communication cost reduction. The need for global communication cost reduction is obvious since sensor networks composed of hundreds to thousands of sensors, generating a remarkable amount of sensor data to be delivered to a data sink [23]. Therefore, efficient architecture(s) and data routing technique(s) are indispensable to reduce global communication cost.

Clustering in a WSN facilitates the underlying flat WSN [3,4] topology and provides a *hierarchical* organization [8-10]. It minimizes communication overhead, increases the probability of aggregating redundant data, as a whole minimizes the overall power consumption [7]. In view of the mobility and scalability, operations such as nodes joining into an existing network and nodes leaving out of an existing network need to be considered [7-9]. Because of the dynamic characteristics of WSN, after a hierarchical clustering has been formed, the maintenance of the cluster becomes very crucial in the presence of network topology changing.

The changes of the physical condition of a WSN lead to the changes of its topology. Once the topology and geography of a WSN change, it becomes necessary to reorganize its network structure and network functions. With regard to mobility and scalability, two topology management operations are considered: node-move-in and node-move-out. Node-move-out and node-move-in are the situations where *nodes are getting out of* and *nodes are joining into* an existing network [7-9]. Even for stationary nodes, when the battery charge is low, it must get out of the network and transition to charge mode. Then, the charged nodes should join back into the network once again. Once a hierarchical clustering is established, the maintenance of the cluster organization becomes crucial in network topology changes.

For example, in the event of a natural disaster in order to identify and collect data of the survivors, to provide medication by establishing a mobile healthcare center, etc. a WSN may be required to deploy rapidly. In such scenario, an efficient WSN deployment technique is a prerequisite which could establish well-organized architecture to facilitate routing of data in a fast and efficient manner.

In [24], we have shown a conceptual idea of data congregation protocols for WSN. However, in this paper, we demonstrate the detail of the protocols and the maintenance algorithms. We also exhibit simulation results to demonstrate the efficiency of the proposed protocols.

In this paper, to facilitate efficient data congregation we construct a cluster-based architecture to a flat Dynamic WSN, where the maintenance of the architecture is done through node-move-in and node-move-out operations [11,12]. We then propose two data congregation techniques where data from all sensor nodes is collected and delivered to a sink node. We also study the time-complexity of the proposed protocols and finally show some simulation results.

Table 1. Summary of our results

Operation	***Time Complexity***
Gathering	$O(p+\Delta)$
Flooding	$O(h)$ and $O(h+\Delta)$
Node-move-in	$O(q)$ and $O(q+h)$
Node-move-out	$O(\|T\|)$ and $O(\|T\|+h)$

Here, p: Number of clusters.
h: Height of $CNet(G)$.
q: Number of neighbours in G of node that wish to join $CNet(G)$.
T: Subtree of $CNet(G)$ rooted by a leaving node *lev*.
Δ: Maximum degree of nodes in G.

2. **Related Works.** In a typical WSN, data gathering protocols facilitate the configuration of the network and collecting information from the desired environment [20]. The main goal of a data aggregation protocol is to gather and aggregate data in an energy efficient manner, i.e., involving as less node as possible so that network lifetime is enhanced. In a WSN, sensor nodes can use different data aggregation techniques to achieve energy efficiency. The objective is to perform an efficient data transmission to any base station (BS) to maximize the lifetime of the network in terms of round, where a round is defined as the process of gathering all the data from sensor nodes to the BS, regardless of how much time it takes. In a data gathering protocol, data from the nodes are required to be collected and transmitted to the nearest BS [21] to make the date accessible for the end user. A simple way of doing that is aggregating (sum, average, min, max, count) the data originating from different nodes [22].

Data congregation has been studied in many literatures [11,12,14-17]. However, to the best of our knowledge only a very few of them have dealt with dynamic cluster-based architecture.

In this paper, we merely concentrate on data gathering and routing of the collected data to a sink node. In our work, we adopt an existing data compression technique presented in [23] and extend the work in terms of efficient transmission of the data to the sink.

In [23], the authors present such a network where sensors are densely deployed in the region of interest and monitor the environment on a regular basis. A routing tree is built where N sensor nodes, denoted as s_1, s_2, ..., and s_N form a multi-hop route to the sink. Then the readings denoted as d_j obtained by node s_j where $j = 1, 2, \dots, N$ are transmitted to the sink through multi-hop relay. Node s_1 transmits its reading d_1 to s_2, and s_2 transmits both its reading d_2 and the relayed reading d_1 to s_3. At the end of the route, s_N transmits all N readings to the sink. The paper then presents a design to apply compressive sampling theory to sensor data gathering for large-scale wireless sensor networks. However, the paper did not present any communication technique to route the data to the sink.

In [11], a cluster-based architecture has been proposed for a Dynamic WSN, where the maximum radius of a cluster is considered to be one or less. Construction and maintenance of a communication highway named *Backbone Tree* (BT) is then defined in order to perform efficient flooding on the network which requires the size of the BT. However, no data congregation protocol on such architecture is proposed. Later on [9], the authors have proposed another architecture with maximum radius of a cluster two or more to achieve better routing. Furthermore, in [10], the authors have presented a better flooding protocol instead of using the size of the BT as a measuring method the authors uses the height of the BT [11]. Again, no data congregation protocol has been proposed.

In [24], the authors presented data gathering protocols for wireless sensor network. However, no maintenance algorithms and simulation results for their protocols have been proposed and also no definition of the cluster-based architecture has been shown.

In this paper, firstly, we propose a dynamic cluster-based architecture and then we show proposed data congregation protocols. Here we propose two congregation protocols, where one takes the benefit of the size of the backbone tree, whereas the other one utilizes the height of it. The proposed first protocol is designed for such scenario where node-move-in and node-move-out operations take place frequently. Whereas, the latter one works better in the scenario where the architecture does not change much once it is established. Secondly, we propose an efficient broadcasting technique to assist our data congregation protocol. Our broadcasting protocol works better than that of presented in [11,12]. Thirdly, we propose maintenance algorithms namely node-move-in algorithm and node-move-out algorithm for the architectures. Finally, we exhibit simulation results for the proposed data congregation protocols and the maintenance algorithms of the cluster-based architecture.

This paper is organized as follows. In Section 3, we define cluster-based architecture. We then demonstrate data congregation protocols in Section 4. In Section 5, we describe node-move-in and node-move-out algorithms for maintenance of the cluster-based architecture. Finally, in Section 6, we exhibit the simulation results before describing conclusion and the future works in Section 7.

3. Network Architecture.

In this section, we describe the assumptions, radio network model, and the cluster-based architecture of a flat wireless sensor network.

Sometimes, a flat wireless sensor network is represented by an undirected graph $G = (V, E)$. Here, V is the set of sensor nodes and E is the set of edges, i.e., links between the sensor nodes. An edge exists between sensor nodes u and v in G, *iff* nodes u and v are in the transmission range of each other.

3.1. The assumptions and radio model.

The following assumptions are made in this paper for a flat wireless sensor network G [11,12,18,19]:

- Nodes have their unique IDs. Prior to joining into a network a node has no knowledge about the network other than its own ID. When a node leaves from the network, all its information other than its own ID is erased.
- In the network, a base station (BS) may exist before a join takes place and the BS is fixed. In other words, there exists at least a node in the network before and after a join or a leave takes place.
- A node repeats transmission/reception, and performs local computation in synchronized fixed *intervals*. In each interval, a node can act either as a transmitter or as a receiver, but not both at the same time. The total number of intervals to collect data to a sink node is called *round*.
- In the network collision detection is not present. In other words, node acting as a receiver in a given interval gets a message *iff* exactly one of its neighbours transmits in this interval. The absence of collision detection is considered to increase the life of a sensor node by reducing the unnecessary transmissions.
- Communication between nodes is symmetric. In other words, all neighbouring nodes can communicate with each other. It means, if a node v can receive a message from a node u then u also can receive a message from v.
- The number of communication channel present in the network is SINGLE.

3.2. **Cluster-based architecture.** The proposed cluster-based architecture is defined below. In this section, we use a graph theoretical approach to define our cluster-based architecture.

Let G be an undirected graph. Nodes in graph G are partitioned into groups, called as clusters. In other words, a cluster is a start subgraph of G. In a cluster, there exists a special node which is the center of the star subgraph called head and is connected to all other nodes called member nodes to form a star topology. Two neighbouring cluster heads are connected via a member node which has an additional role of a gateway node (Figure 1).

Let graph $G = (V, E)$ to be an undirected graph with a specified node r with n $(n \geq 1)$ nodes. Then a *cluster-based network* of G, denoted as $CNet(G)$ is a spanning tree of G with root r. In $CNet(G)$, a node knows its status, i.e., either as a *cluster-head,* or as a *gateway*, or as a *member*, whereas the root r is a cluster-head.

In Figure 1, we exhibit a cluster-based architecture $CNet(G)$ obtained from a flat sensor network G, where red nodes are cluster-heads, blue nodes are gateway nodes and green nodes are member nodes. One special head node is designated as the root (denoted as r in Figure 1(b)) in the architecture.

Given a graph $G = (V, E)$. Let $CNet(G) = (V, E_{CNet(G)})$ be its cluster-based network and $G^{'} = (V \cup \{new\}, E \cup E^{'})$ be a graph obtained after a new node *new* is added to G. In G', $E' = \{(new, u) | u \in V\}$, where *new* and u are in each other's transmission range. A cluster-based network of G' is defined as $CNet'(G) = (V \cup \{new\},\ E_{CNet} \cup \{(new, w)\})$. $CNet'(G)$, w is the parent of *new.*

In a cluster-based network $CNet(G)$, each node has its own status. The status of a new node *new* on *the* $CNet'(G)$ is decided as per the following rules. Given a set of nodes U, where U is connected to *new* and w is the node that is to be selected by *new.* In G', the nodes have the same status as they have in $CNet(G)$, except for nodes *new* and w. In U, if there exist cluster-head(s), then *new* selects one as w and becomes member of w. Otherwise, in U if there exist gateway(s), then *new* selects one as w and *new* becomes cluster-head of a new cluster. Else, in U there exist only member nodes. In such case, *new* selects one as w; then set w to be a gateway and *new* becomes cluster-head of a new cluster.

A backbone of $CNet(G)$ is formed by cluster-heads and gateways and denoted as $BT(G)$. In other words, $BT(G)$ is a sub-tree of $CNet(G)$ formed by cluster-heads and gateways. Figure 2 gives an example of the formation of cluster-based network.

We describe below the properties of a cluster-based network $CNet(G)$.

Property 3.1. [5] According to the rules of cluster formation (to reduce the number of clusters), no two cluster-heads can be neighbours with each other in graph G. To strictly follow this strategy, if p is the smallest number of complete sub-graph in G, then $CNet(G)$ has at most p number of clusters. Moreover, in $CNet(G)$, a $BT(G)$ is formed by cluster-heads and gateway nodes. Since a gateway node is connecting two cluster-heads there are at most $2p - 1$ nodes in $BT(G)$ (one gateway node less for the root cluster).

Property 3.2. To strictly follow the rule that no two cluster-heads can be neighbours with each other in graph G, there can be at most 5 cluster-heads in its neighbors. Otherwise, there will be at least two cluster-heads connected to each other in G.

Property 3.3. Let G be a unit disk graph, $CNet(G)$ be its cluster-based network and $BT(G)$ be the backbone tree of $CNet(G)$. Then the maximum degree of $BT(G)$ is 19, i.e., it is a constant value.

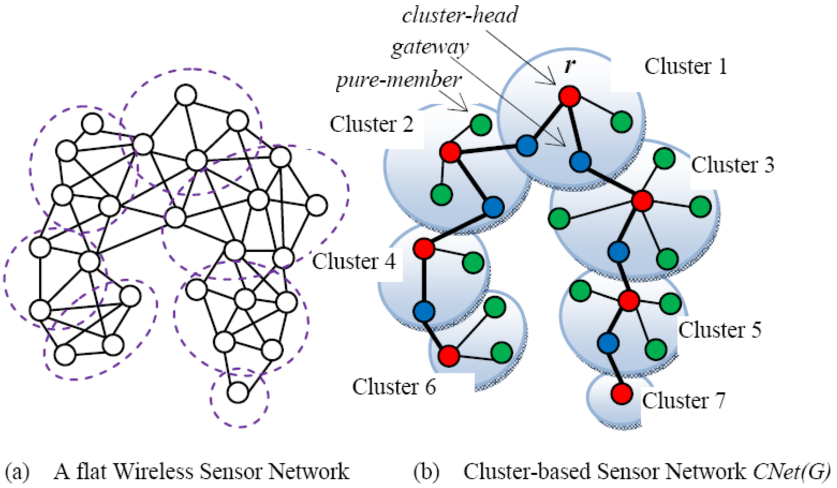

Figure 1. A cluster-based wireless sensor network $CNet(G)$ is derived from a flat wireless sensor network

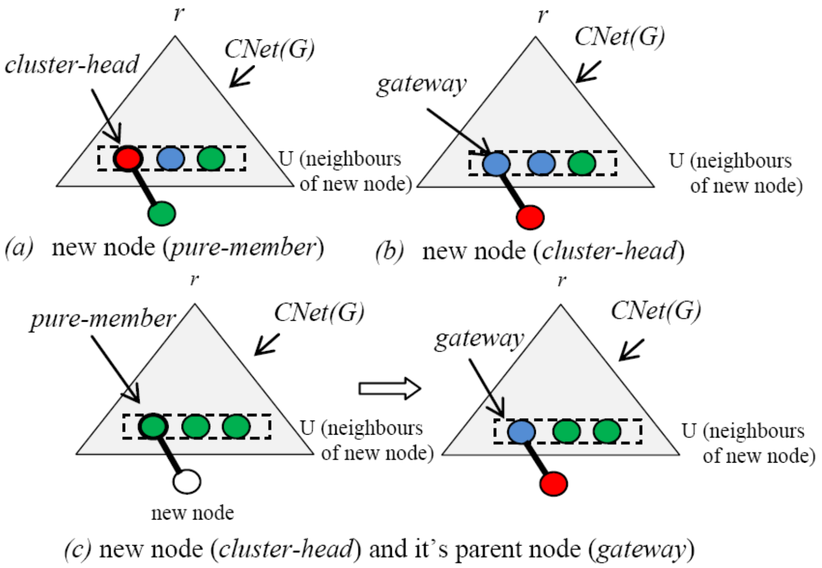

Figure 2. Formation of a cluster-based architecture $CNet(G)$

In $BT(G)$, gateway nodes are adjacent to the cluster-heads. Moreover, no two cluster-heads are to be allowed to be neighbors with each other and there can be at most 5 cluster-heads as neighbor of a node. Therefore, the degree of gateway nodes is less than 6.

In addition, cluster-heads are adjacent to gateway nodes only and gateway nodes connect two or more cluster-heads on $BT(G)$. Therefore, the degree of a cluster-head h on $BT(G)$ is less than or equal to the number of cluster-heads within 2-hop from h. Since the number of cluster-heads within 2-hop is less than 20 [13], the maximum degree of $BT(G)$ is at most 19.

In [11], a definition of *Eulerian* $BT(G)$ is given where a node in $BT(G)$ transmits a message at least once and in each interval exactly one node transmits a message. We adopt *Eulerian* $BT(G)$ for one of our congregation protocols. In *Eulerian* $BT(G)$, a message travels an *Eulerian* tour on a $BT(G)$, i.e., a message (is often called as a token) starts travelling from a source node and visits every node before returning to the source node. At the beginning, the token resides in the source node and starts visiting each node in $BT(G)$ starting from the source node in depth-first order. When a node u in $BT(G)$ receives the token, u sends the token and its ID with the message to one of its neighbours that has not received the token yet. In the case that u has no neighbour left which it has not been visited by the token yet, it returns the token to the node from which has been received for the first time. On this tour, the rotation of the token forms an *Eulerian* cycle of $BT(G)$ where it travels every node in $BT(G)$ and returns to the source node.

A cluster-graph $CGraph(G)$ is constructed from $CNet(G)$ where each cluster-node in the graph represents a cluster of $CNet(G)$. In $CGraph(G)$, there exists a cluster-edge between two clusters. For example, let $C1$ and $C2$ be two clusters and there exists at least a node $u \in V(G)$ s.t. it is connecting the cluster-heads of $C1$ and $C2$.

Constructing cluster-graph s.t. no two neighbouring cluster nodes have the same color this way and using the property of $CNet(G)$, we can conclude that the total number of colors required to color $CGraph(G)$ is at most 20.

In a node-move-in operation, a node *new* moves into an existing $CNet(G)$ and the network is re-organized by adding *new*. In a node-move-out operation, a node *lev* leaves from the existing $CNet(G)$ and the network is re-organized by removing *lev* from G.

Initially nodes in G know their IDs. Later, the following information is maintained at each node through node-move-in and node-move-out operations:

- Self ID.
- Self status, i.e., cluster-head, gateway, or member IDs and status of all 1-hop neighbours in G.
- Parent's ID.

4. Data Congregation Protocols.

In this section we describe our proposed data congregation protocols GPR_1 and GPR_2. GPR_1 performs better where node-move-in and node-move-out operations take place frequently. Whereas, GPR_2 works better where $CNet(G)$ does not change frequently once it is established.

In both GPR_1 and GPR_2 data are to be gathered to a sink node denoted as *sink*. Let $h(sink)$ be the cluster-head of *sink*. Proposed data congregation protocols are described below.

4.1. Data congregation protocol GPR_1.

The proposed GPR_1 protocol works in three steps.

In the first step, the sink node disseminates a message to the rest of the nodes in the network that is requesting to be ready for data congregation. In the case that the sink node is not a cluster-head then the cluster-head initiates this step.

In the second step, data is gathered in each cluster.

In the third step, gathered data are transmitted to the sink node (see Figure 3).

The detail of the protocol is described below.

Step 1: *sink* calls $Eulerian(BT(G))$. Through this Eulerian the size of p, i.e., the number of clusters is known to the *sink*. However, if the *sink* is not a cluster-head, $Eulerian(BT(G))$ is called by $h(sink)$, i.e., by the cluster-head of the *sink*.

Step 2: The *sink* node (or it is cluster-head $h(sink)$, if the sink is not a cluster-head) generates a message (gpr, $slot$, p, Δ) and performs $Eulerian(BT(G))$. Upon receiving the message, each cluster-head calculates intervals using the formula *4p-2-slot* and waits for the current *Eulerian* to finish. Here *slot* is the current time-slot, p is the number of clusters in the network and Δ is the maximum degree of a cluster-head in $CNet(G)$. We will call Δ as maximum degree in $CNet(G)$.

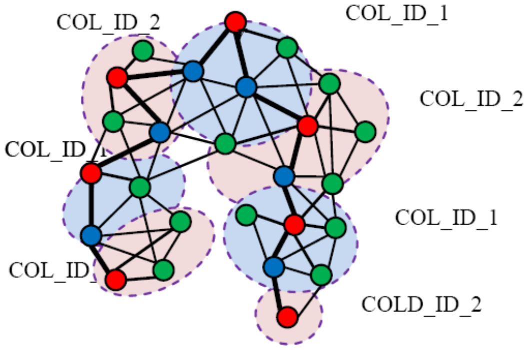

FIGURE 3. Coloring technique used for data congregation technique in GPR_1

Once *Eulerian*$(BT(G))$ is finished, each cluster-head in $CNet(G)$ having the j-th COL_ID (where $1 \leq j \leq 20$) starts the congregation of data from its member nodes in the $(j*\Delta)$-th interval.

Step 3: *sink* (or its cluster-head $h($*sink*$)$) calls a final *Eulerian*$(BT(G))$ to collect the gathered data from each cluster-head of *CNet*(G).

Theorem 4.1. *Let G be a graph, $CNet(G)$ be the cluster-based network, and $CGraph$ (G) is the cluster-graph. Then GRP_1 can be completed in $O(p+\Delta)$ intervals, where, p is the number of clusters in $CNet(G)$ and Δ is the maximum degree in $CNet(G)$.*

Proof: Since there are $2p-1$ nodes in $BT(G)$ and Δ is the maximum number of nodes in a cluster of $CNet(G)$, each $Eurelian(BT(G)$ requires $O(p)$ intervals and congregation inside the cluster-heads requires at most $20*\Delta$ intervals. Hence, the total number of intervals required in this process is $O(p+\Delta)$.

4.2. Data congregation protocol GPR_2.

Here the construction of $CNet(G)$ is further improved s.t. efficient flooding can be performed using the height of the architecture instead of the size (which is done in GPR_2). Principally, in flooding, we use a similar approach presented in [12].

In $CNet(G)$, let $h_{CNet(G)}$ be the height of $CNet(G)$ and Δ be the maximum degree of nodes in $CNet(G)$. Then each node in $CNet(G)$, resides in a level, e.g., *level* 0, *level* 1, …, *level* i, *level* $i+1$, …, *level* $h_{CNet(G)}$, and the *root* resides in the lowest level, i.e., *level* 0. The root knows $h_{CNet(G)}$ and Δ that are updated during each node joins into $CNet(G)$ and are maintained after a node leaves (see Figure 4).

The proposed congregation protocol GPR_2 can be mainly divided into two steps.

In the first step, *sink* communicates with the root r and asks to initiate flooding to inform all other nodes about a data congregation in order to notify related information such as the height and the maximum degree in $CNet(G)$.

Whereas, in the second step, a reverse flooding is performed where data is aggregated in each cluster and then delivered to their parents in the upper level. In the first step, we use the flooding technique presented in [12]. Here flooding starts from the root and nodes at *level* $i+1$ participate in transmission *iff* nodes in *level* i have already completed transmissions. On the other hand, in the second step, flooding technique including the data aggregation technique presented before is used.

Here we adopt two coloring techniques which have also been presented in [18]: one is to forward data between clusters residing in the neighbouring levels of the cluster-graph (for details see [18]), we call here nodes with COL_ID_i $(1 \leq i \leq 20)$ and the other one is

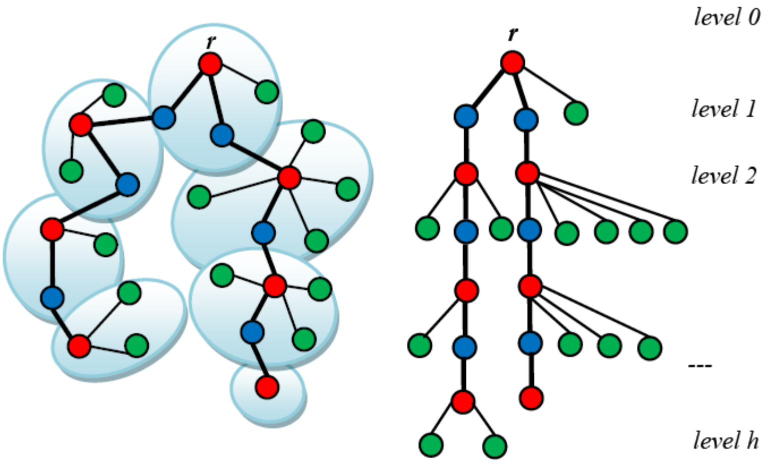

FIGURE 4. A cluster-based sensor network, where in each level there exist only cluster-heads or gateway and member nodes

to transmit the message inside the cluster which is used for data, we call here nodes with COL_ID_j ($1 \leq j \leq 20$).

We present below proposed data congregation protocol GPR_2:

Step 1: If the *sink* is not the root it asks the root to initiate data congregation.

Step 2: The *root* generates a message (*gpr*, *slot*, h, Δ) and floods the message.

Step 3: Once the flooding ends at *level* $h_{CNet(G)}$, nodes at *level* $h_{CNet(G)} - 1$ with COL_ID_j start aggregation in (j*Δ)-th interval and wait all aggregation in this level to be finished.

In the aggregation process, a parent node in *level* i asks its children residing in *level* $(i + 1)$ to transmit data one-by-one.

Step 4: Once aggregation at a level is completed node with COL_ID_i forwards the data to its upper level in the i-th interval, i.e., a node at *level* i forwards the data to its parent at *level* $(i - 1)$.

Step 5: Finally, *root* forwards the gathered data to the *sink*.

Theorem 4.2. *Let G be a graph, $CNet(G)$ be its cluster-based network, and $CGraph(G)$ be the cluster-graph. Then data congregation protocol* GPR$_2$ *requires $O(h + \Delta)$ intervals to collect data to* sink, *where h is the height of the cluster-based architecture $CNet(G)$ and Δ is the maximum degree in graph G.*

Proof: To send the request from the sink to the root and to receive back the aggregated data it requires $O(h)$ intervals. The flooding requires $O(h)$ intervals whereas the aggregation requires $O(\Delta)$ intervals. Thus the total number of intervals required in this process is $O(h + \Delta)$.

5. **Maintenance Algorithms.** This section describes maintenance algorithms for two operations *node-move-in* and *node-move-out*. According to the definition of *CNet*(G) in Section 2, each node in *CNet*(G) needs to have the following knowledge:

- It knows its neighbours in G and *CNet*(G), and the parent in *CNet(G),* respectively. It knows its status (as a cluster-head or a gateway node or a pure-member).
- Each cluster-head maintains its colour ID.
- The root node knows $h_{CNet(G)}$ and Δ.
- Each cluster-head and gateway node maintains its transmitting time slot.
- Each node maintains its level ID.

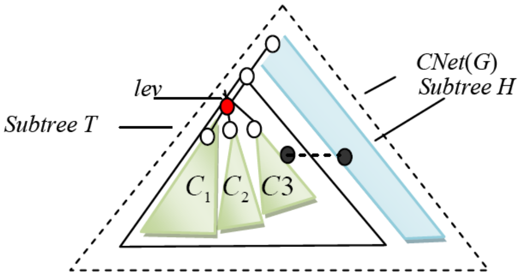

FIGURE 5. $CNet(G)$ is divided into two subtrees T and H

In [7,9], for constructing and reconfiguring *CNet*(G), the nodes of *CNet*(G) maintain knowledge (i) only. It is done as follows. A new node sends "Please allow me to join" message to its neighbors. Upon hearing it nodes *CNet*(G) who have received the message call SIMULATE-IN procedure and send their IDs and status one by one. Once the new node knows about all of its neighbors in *CNet*(G) it determine its status as described in Section 3.2.

To maintain knowledge of colour ID, $h_{CNet(G)}$ and Δ few more steps are required which are described in the following section.

5.1. **Node-move-in algorithm.**

We add two additional phases after the node-move-in operation of [7] as follows:

Step 1: After determining its own status, node new informs its neighbouring clusters about its ID, status and own neighbouring cluster(s)-head's ID(s) one by one.

If new finds the cluster-head(s) in its neighbour, new chooses the one with a smaller level ID and informs of it directly.

Else if there are gateways in its neighbours that are connected to the cluster-head, *new* chooses the gateway node with lowest ID and level ID.

Else new chooses the pure-member with the lowest ID and level ID that is connected with the cluster-head.

Step 2: Upon receiving information from node *new*, each neighbouring cluster-head h then updates its information.

Theorem 5.1. *Let $CNet(G)$ be a cluster-based network of G, then joins of* new *into $CNet(G)$ can be done in expected $O(d+h)$ intervals, where d is the number of neighbours of the new node* new *and h is the height of the cluster-based network.*

Proof: According to Theorem 3 of [5], it requires expected $O(d)$ intervals to collect neighbors' information and to determine the status and parent node of *new*, i.e., knowledge (i). In order to achieve knowledge (ii) we use Property 3.3. According to the property in the cluster-based structure a node can have at most 19 cluster-heads in its 2-hop neighbours. Thus, *new* requires 19 more intervals to inform the neighbouring clusters' information to those cluster-heads. Then to update the information on potential nodes, it requires at most 19 more intervals. Thus, the whole processes here can be done in $O(1)$ intervals.

5.2. **Node-move-out algorithm.**

Let graph $G = (V, E)$ have n ($n \geq 1$) nodes and $CNet(G) = (V, ECN_{et(G)} = (V, E_{CNet})$. A graph obtained by deleting a node Lev from G is a graph, where $E' = \{(lev, u)|(lev, u) \in E\}$. We assume that the graph G is connected and after a leave the resulting graph is also connected.

We divide $CNet(G)$ into two sub-trees: the tree T with *lev* as the root, and the tree H whose root is the root of $CNet(G)$ (when the node *lev* is the root of $CNet(G)$ can be dealt similarly.

Assume that C_i $(i = 1, 2, 3, \dots)$ are the sub-trees of *lev* in T. Since G is connected, after *lev* leaves, there exists at least one edge e in G which is neither an edge of T nor an edge of G but connects a node u of T with a node v of H. In [7], $CNet(G)$ with knowledge (i) is reconfigured in $O(|T|)$ intervals by using the following two phases:

Step 1: *lev* calls *Eulerian*(T). Here, a message "Find an edge that is not in T" is sent to find the edge $e = (u, v)$.

Step 2: node u calls *Eulerian*$(T - \{lev\})$. Here a message "Join into H" is sent to start an *Eulerian* tour in $T - \{lev\}$ from node u until all the nodes of $T - \{lev\}$ moved into H.

However, to maintain knowledge (ii) the following additional phase is performed:

Step 3: finally, once the new clustering is formed, node u' calls *Eulerian*(T'), where u' is the node that found the edge with H and T' is the subtree rooted by u excluding *lev*. In this procedure each node in T' updates their neighbouring cluster and potential nodes' information as in the *Node-Move-In* algorithm.

In our node-move-out operation, we need to maintain knowledge (ii) too. Before moving the nodes of T into H, the nodes of H need to delete the nodes of T from their neighbours' lists and recalculate their neighbouring clusters and potential nodes' information.

Theorem 5.2. *Let $CNet(G)$ be a cluster-based network of G, then leave of lev from $CNet(G)$ can be done in $O(|T| + h)$ intervals, where T is the subtree rooted by the leaving node lev and h is the height of $CNet(G)$.*

Proof: Using our Node-Move-In algorithm and Theorem 4 of [5] we can prove the theorem. In the first phase, node *lev* calles *Eulerian*(T) by which each node in T can know about its presence in T and whether there exists any neighbouring node in H. This requires $O(|T|)$ intervals. In the second phase, $Eulerian(T')$, where $T' = T - \{lev\}$, is called by which nodes in T other than *lev* moves in H to determine their status and parents which also take $O(|T|)$ intervals. Finally, in the third phase, $Eulerian(T')$ is called by which nodes in T' update their information on neighbouring clusters and select potential nodes to the neighbouring cluster. Since according to Property 3.3 each node has at most 19 cluster heads, i.e., clusters in its neighbours according to node-move-in algorithm this process requires $O(|T|)$ intervals. Moreover, h rounds are required to update $h_{CNet(G)}$ and Δ in the root.

Hence, the total number of intervals are required for Node-Move-Out algorithm is $O(|T| + h)$.

6. **Simulation Results.**

In this section, simulation results are presented. The experimental environment builds on Java language. The experiments are carried on random unit disk graphs that are generated in 1000m × 1000m square fields. The transmission range of each node is set to 50m. n number of nodes are used in the experiment varying from 200 to 1000. For each number of nodes the experiments are repeated 100 times, each time by generating a random unit disk graph. The average results are then presented here.

Figure 6 shows the cluster information in the network after n number of nodes form a cluster-based network. Initially it could be easily seen an increased number of clusters when the number of nodes in the network is small. However, as long as the network grows larger the number of clusters decreases proportionally, thus shows the effectiveness of the

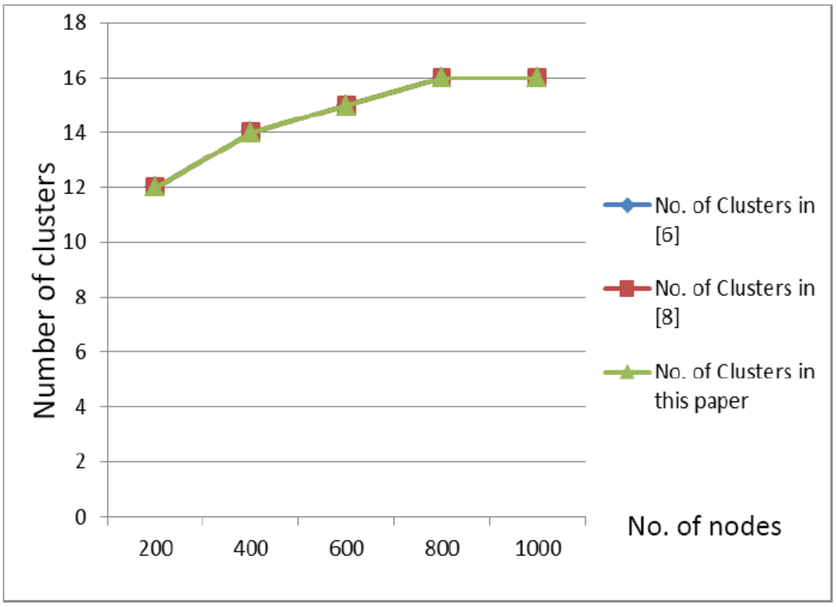

Figure 6. Number of clusters in the cluster-based architecture $CNet(G)$

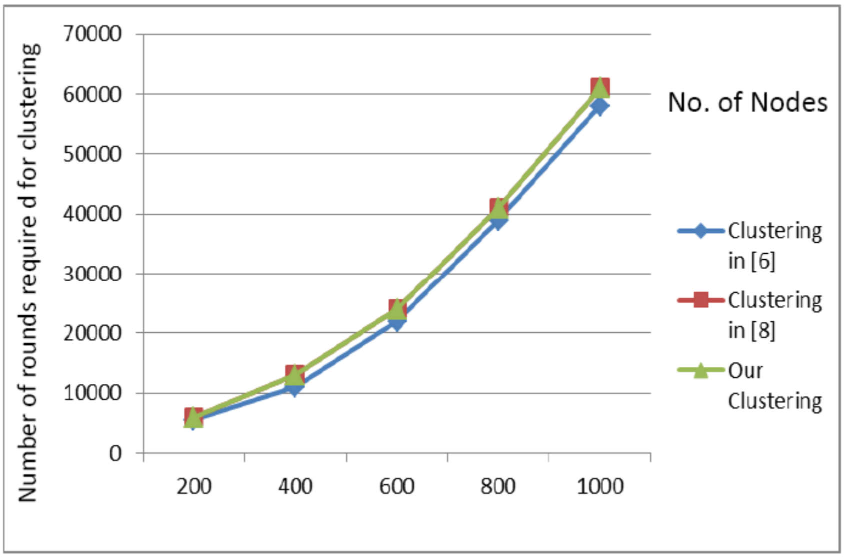

Figure 7. Computational complexities of forming cluster-based networks

proposed model. It is observed that using the proposed clustering the number of clusters does not change compared within [6,8].

In Figure 7, the computational complexities of forming the network are presented. Here a comparison between computational time to form a cluster-based network using the proposed model and the techniques proposed in [6,8] is presented. It is observed that our proposed model uses less computational time than [8]. However, it has a bit higher computational time than [6]; this is because once a node joins an existing network it does not have to communicate with the root node to update the height of the network.

In Figure 8, we show a comparison of the effect on the number of clusters before and after leave of a cluster-head and/or gateway node. It is observed that in most of the cases the number of clusters does not change much.

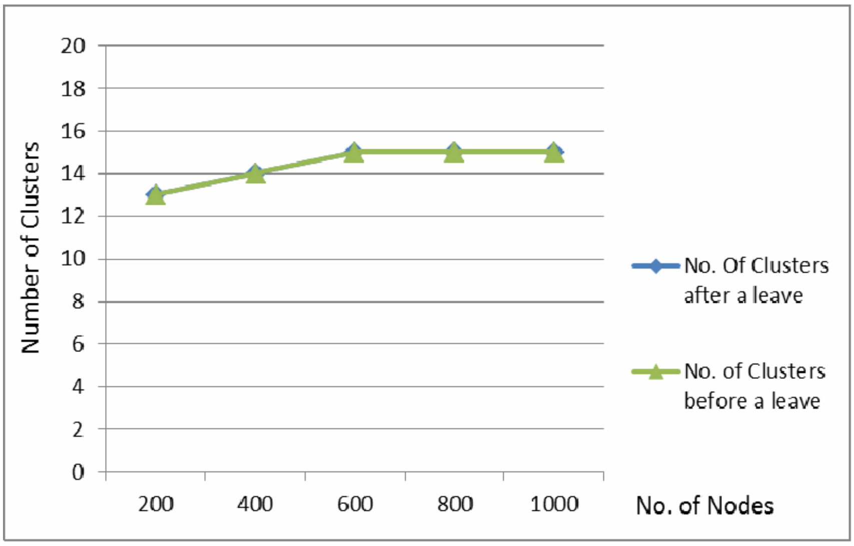

Figure 8. Number of clusters after leaving a node from the network

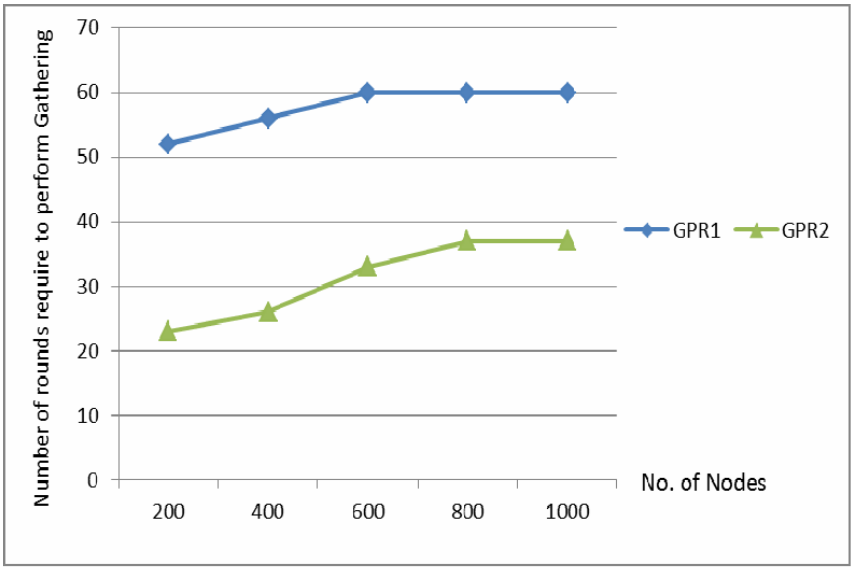

Figure 9. Number of intervals require in GPR_1 and GPR_2

Finally, Figure 9 depicts the Computational complexities of gathering protocols GPR_1 and GPR_2. It is observed that GPR_2 protocol uses less computational time than the GPR_1.

7. Conclusion and Future Work.

In this paper, we have presented two efficient data gathering protocols for dynamic cluster-based wireless sensor networks, depending on the needs.

First, a Dynamic Cluster-based wireless sensor network has been presented to support efficient flooding. Then flooding protocols on this architecture have been presented. Finally, experiments have been made where the simulation results showed that the proposed flooding protocols gave better performance than some protocols in a similar architecture.

In future work, we would like to concentrate on the following aspects in a similar dynamic cluster-based wireless sensor network model:

Firstly, we would like to design a cluster-based structure with multiple simultaneous node-move-in and node-move-out operations. We also plan to reduce the time complexity for a join and a leave operation.

Secondly, we would like to establish a method for our dynamic cluster-based structure with which nodes could communicate with each other in a secured manner.

Thirdly, we plan to propose new architectures with better properties than that of the architecture $CNet(G)$ in this paper.

In addition, we intend to consider fault-tolerance and self-stability. Fault-tolerance is necessary because of the instability of both the node itself and of the communication via radio. We comprehend that the achievement of fault-tolerance is as important as our development progresses in the future. Self-stabilization is considered to be a promising part of that. One of the most important goals in achieving self-stabilization is to get rid of the assumption that the node joins one by one from an initial state (one node), which is our present model. Therefore, it is necessary to consider clustering from an arbitrary situation.

Lastly, we would like to consider not only the problems with this network communication model but also the validity of the model itself in order to bring it closer to reality.

Acknowledgment. This work is partially supported by a grant with Reference No. 4J044 of Malaysia-Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia (UTM) and MOHE. The authors gratefully acknowledge the useful comments and suggestions of all reviewers in the ISUVR2011 International Conference, South Korea and the reviewers of International Journal of Innovative Computing, Information and Control (IJICIC) that have helped the authors to improve the presentation of the paper in its current form. Finally, we would like to dedicate this work to Late Professor Datuk Dr. Marzuki Bin Khalid, former Deputy Vice-Chancellor, UTM, Malaysia.

REFERENCES

[1] Y. Obashi, H. Chen, H. Mineno and T. Mizuno, An energy-aware routing scheme with node relay willingness in wireless sensor networks, *International Journal of Innovative Computing, Information and Control*, vol.3, no.3, pp.565-574, 2007.

[2] C.-C. Chiang, Routing in clustered multihop, mobile wireless networks with dading channel, *Proc. of IEEE SICON*, pp.197-211, 1997.

[3] J. Yick, B. Mukherjee and D. Ghosal, Wireless sensor network survey, *Computer Networks*, vol.52, no.12, pp.2292-2330, 2008.

[4] G. Simon, M. Maroti, A. Ledeczi, G. Balogh, B. Kusy, A. Nadas, G. Pap, J. Sallai and K. Frampton, Sensor network-based countersniper system, *Proc. of the 2nd International Conference on Embedded Networked Sensor Systems (Sensys)*, Baltimore, MD, USA, 2004.

[5] J. Yick, B. Mukherjee and D. Ghosal, Analysis of a prediction-based mobility adaptive tracking algorithm, *Proc. of IEEE the 2nd International Conference on Broadband Networks (BROADNETS)*, Boston, 2005.

[6] M. Castillo-Effen, D. H. Quintela, R. Jordan, W. Westhoff and W. Moreno, Wireless sensor networks for flash-flood alerting, *Proc. of the 5th IEEE International Caracas Conference on Devices, Circuits, and Systems*, Dominican Republic, 2004.

[7] T. Gao, D. Greenspan, M. Welsh, R. R. Juang and A. Alm, Vital signs monitoring and patient tracking over a wireless network, *Proc. of the 27th IEEE EMBS Annual International Conference*, 2005.

[8] K. Lorincz, D. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh and S. Moulton, Sensor networks for emergency response: Challenges and opportunities, *Pervasive Computing for First Response (Special Issue), IEEE Pervasive Computing*, 2004.

[9] G. Wener-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees and M. Walsh, Deploying a wireless sensor network on an active volcano, *Data-Driven Applications in Sensor Networks (Special Issue), IEEE Internet Computing*, 2006.

[10] S. Basagni, M. Mastrogiovanni and C. Petrioli, A performance comparison of protocols for clustering and backbone formation in large scale ad hoc networks, *The 1st International Conference on Mobile Ad-Hoc and Sensor Systems*, pp.70-79, 2005.

[11] J. Uchida, A. K. M. Muzahidul Islam, Y. Katayama, W. Chen and K. Wada, Construction and maintenance of a novel cluster-based architecture for ad hoc sensor networks, *Journal of Ad Hoc and Sensor Wireless Networks*, vol.6, no.1-2, pp.1-31, 2008.

[12] W. Chen, A. K. M. Muzahidul Islam, M. Malkani, A. Shirkhodaie, K. Wada and M. Zein-Sabatto, Novel broadcast/multicast protocols for dynamic sensor networks, *IEEE International Parallel & Distributed Processing Symposium*, USA, 2007.

[13] M. Goldberg, Packing of 14, 16, 17 and 20 circles in a circle, *Mathematics Magazine*, vol.44, no.3, pp.134-139, 1971.

[14] K. Ramanan and E. Baburaj, Data congregation algorithms for wireless sensor networks: A survey, *International Journal of Ad Hoc, Sensor & Ubiquitous Computing*, vol.1, no.4, 2010.

[15] R. Bar-Yehuda, O. Goldreich and A. Itai, On the time-complexity of broadcast in radio networks: An exponential gap between determinism and randomization, *Journal of Computer and System Science*, vol.45, pp.104-126, 1992.

[16] P. Mohanty, S. Panigrahi, N. Sarma and S. S. Satapathy, Security issues in wireless sensor network data congregation protocols: A survey, *Journal of Theoretical and Applied Information Technology*, pp.14-27, 2010.

[17] B. S. Chlebus, L. Gasieniec, A. M. Gibbons, A. Pelc and W. Rytter, Deterministic broadcasting in ad hoc radio networks, *Distributed Computing*, vol.15, pp.27-38, 2002.

[18] A. K. M. Muzahidul Islam, S. Baharun, S. S. Abdullah, R. Yusof, W. Chen and K. Wada, A dynamic WSN structure for network cloud and an efficient flooding technique, *ICMSAO'11 the 4th International Conference on Modeling, Simulation & Applied Optimization*, Malaysia, 2001.

[19] A. K. M. Muzahidul Islam, K. Wada, J. Uchida and W. Chen, A better dynamic cluster-based structure wireless sensor network for efficient routing, *International Journal of Innovative Computing, Information and Control*, vol.8, no.10, pp.6747-6760, 2012.

[20] P. Mohanty, S. Panigrahi, N. Sarma and S. S. Satapathy, Security issues in wireless sensor network data gathering protocols: A survey, *Journal of Theoretical and Applied Information Technology*, 2010.

[21] J. Norman, J. P. Joseph and P. P. Roja, A faster routing scheme for stationary wireless sensor networks – A hybrid approach, *International Journal of Ad Hoc, Sensor & Ubiquitous Computing*, vol.1, no.1, pp.1-10, 2010.

[22] M. R. E. Jebarani and T. Jayanthy, An analysis of various parameters in wireless sensor networks using adaptive FEC technique, *International Journal of Ad Hoc, Sensor & Ubiquitous Computing*, vol.1, no.3, pp.33-43, 2010.

[23] C. Luo, F. Wu, J. Sun and C. W. Chen, Compressive data gathering for large-scale wireless sensor networks, *MobiCom'09*, Beijing, China, 2009.

[24] A. K. M. Muzahidul Islam, W. Chen and K. Wada, Efficient data gathering protocols for wireless sensor networks, *The 6th International Symposium on Ubiquitous Virtual Reality*, Korea, 2011.

[25] C.-L. Wang, T.-P. Hong, G. Horng and W.-H. Wang, A GA-based key-management scheme in hierarchical wireless networks, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(A), pp.4693-4702, 2009.

[26] M. A. Khan, G. A. Shah and M. Sher, A QoS based multicast communication framework for wireless sensor actor networks (WSANs), *International Journal of Innovative Computing, Information and Control*, vol.7, no.12, pp.7003-7020, 2011.