International Journal of Innovative
Computing, Information and Control
Volume **8**, Number **10(A)**, October **2012**

ICIC International ©2012 ISSN 1349-4198
pp. **6747–6760**

A BETTER DYNAMIC CLUSTER-BASED STRUCTURE OF WIRELESS SENSOR NETWORK FOR EFFICIENT ROUTING

A. K. M. Muzahidul Islam[1,*], Koichi Wada[2], Jiro Uchida[3] and Wei Chen[4]

[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
Faculty of Science and Engineering
Hosei University
3-7-2 Kajino-cho, Koganei, 184-8584, Japan
wada@hosei.ac.jp

[3]Software Development Division
AISIN AW Co., Ltd.
Okazaki, Japan
cepter1218@ybb.ne.jp

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

Received May 2011; revised December 2011

Abstract. *In recent years, wireless sensor networks have gained a tremendous amount of attention due to their potential ability in providing solutions in various areas such as health care, environment, defense, surveillance, industry and transport. Typically, the sensors are small, with limited processing and computing resources and thus useful for network operations. In this paper, we present an improved Dynamic Cluster-based Wireless Sensor Network (WSN) that facilitates an efficient routing protocol. The cluster-based structure presented here is self-constructible and reconfigurable and is supported by two atomic operations: node-move-in and node-move-out. Our routing protocol finds routes on graph G, unlike some previous routing protocols that find routes on the structure in a similar cluster-based structure. For the two operations we also propose two algorithms: Node-Move-In and Node-Move-Out. We show that to establish a route on graph G using the structure, it requires $O(p)$ rounds, where p is the number of clusters in the network. Note that, in a scenario where the number of sensor nodes n is enormous, p is much less than n. We also show that the proposed Node-Move-In and Node-Move-Out algorithms require expected $O(q)$ and $O(|T|)$ rounds, respectively. Here q is the number of neighbors in G of the node that wish to join to an existing cluster-based structure and T is the sub-tree of the structure whose root is the leaving node. Finally, our simulation results describe that the proposed routing protocol finds a better route with less length and using less computational time.*

Keywords: WSN, Routing, Node-move-in and node-move-out, Dynamic cluster-based WSN

1. **Introduction.** A Wireless Sensor Network (WSN) contains hundreds or thousands of sensor nodes that have the ability to communicate either among each other or directly with an external base-station. In the last few years, Wireless Sensor Networks (WSNs), the dense wireless networks of sensor nodes collecting and disseminating environmental data,

have received a tremendous amount of attention in the research field. There are many 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 [1,16]. The sensor nodes 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]. Clustering is used to leverage the underlying flat sensor network topology and provide a hierarchical organization [9-11].

The topology of a WSN changes when its physical condition is changed. When the topology and geography of WSN changes, its network structure and network functions need to be reorganized. 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 [5-7]. 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.

The underlying objective of any routing protocol is to render the network useful and efficient. A routing protocol coordinates the activities of individual nodes in the network to achieve global goals and to do so in an efficient manner [3,4,6,12-14]. To minimize communication overhead and facilitate energy efficient routing we construct a cluster-based structure within a flat dynamic WSN, where the maintenance of the structure is done through node-move-in and node-move-out operations [5-7].

Our proposed cluster-based architecture and the routing technique could be applied to a scenario when sudden natural disaster occurs. In this situation, sensor nodes would be deployed physically in such a way that they form a dynamic cluster in order to monitor a network. The sensor nodes then can collect critical data from the network (e.g., information on survivors) and forward the message efficiently to gateway computer nodes using the proposed routing technique.

In [5], a novel cluster-based structure for a dynamic WSN is presented, where the maximum radius of a cluster is one. To perform efficient broadcasting the structure then constructs and maintains a Communication Highway called Backbone Tree (BT). Broadcasting is done on this structure using the size of the BT. However, no routing protocol for this structure is proposed.

Later, in [6], to find a routing path better than the size of BT, the authors have proposed another novel cluster-based structure where the maximum radius of clusters is more than one. To perform efficient routing a Communication Super Highway called Super Backbone Tree (SBT) is developed which is smaller than BT in size. Then a route on SBT is created. Finally, it is shown that to find a route on graph G it requires $O(n)$ rounds (where n is the number of the nodes in the network), which is the gossiping time on this network. The process requires that each node in the network participates in finding a shortest path on G, which is not realistic for a WSN where the nodes are memory, processing power and energy constrained.

However, communicating to the nodes using BT and/or SBT only is not realistic. This is because the nodes in the network have only 1-hop data (some cluster head nodes in SBT have at most r-hop (where $r \geq 2$) node information). Therefore, to communicate to a node which is a 3-hop neighbour, the source node has to traverse completely through the path of BT or SBT each time to get into the destination node.

To solve this problem, in this paper, we propose to first re-design the cluster-based structure and then we propose an efficient routing protocol that uses the improved structure to find a path on graph G. The most important benefit of this approach is that once the route is developed, each packet uses this path to reach the destination node thus making it much more efficient than some previous routing approaches on a similar structure, where each time a packet has to traverse completely the cluster-based structure to reach the destination node. We also study maintenance of the cluster-based structure where we give two algorithms: for node-move-in operation we propose Node-Move-In algorithm and for node-move-out we present Node-Move-Out algorithm.

First, we describe the improved cluster-based structure in Section 2. Then, in Section 3, we present our routing protocol. Here we also show that to create a route on graph G using the cluster-based structure it requires $O(p)$ rounds, where p is the number of clusters. In Section 4, we present the maintenance algorithms for the cluster-based structure. We describe that the proposed Node-Move-In and Node-Move-Out algorithms can be done in expected $O(q)$ and $O(|T|)$ rounds, respectively. Here q is the number of neighbors in G of the node that wish to join to a existing cluster-based and T is the sub-tree of the structure whose root is the leaving node. In Section 5, we present some simulation results of our proposed routing protocol. Finally, in Section 6, we present the conclusion of the paper. Table 1 gives the summary of our results.

TABLE 1. Summary of our results

Operation	***Our Results***	***Some Previous Results***
Routing Path	*On Graph G*	*On Cluster-based Architecture*
Routing Time	$O(\|BT\|)$	$O(\|SBT\|)$
Node-move-in	$O(q)$	$O(q+k)$, *where* $k \geq 2$
Node-move-out	$O(\|T\|)$	$O(\|T\|k)$, *where* $k \geq 2$

Here, p is the number of clusters in $CNet(G)$, q is the number of neighbors in G of the node that wishes to join $CNet(G)$, and T is the sub-tree of the cluster-based structure whose root is the leaving node.

2. **Cluster-based Structure for Dynamic Wireless Sensor Networks.** In this section, we describe the model and the cluster-based structure for a flat wireless sensor network.

A wireless sensor network can be represented by an undirected graph $G = (V, E)$, where V is the set of nodes that represent the sensor nodes and E is the set of edges. In graph G, nodes u and v have an edge between them if and only if they are in each others' transmission range.

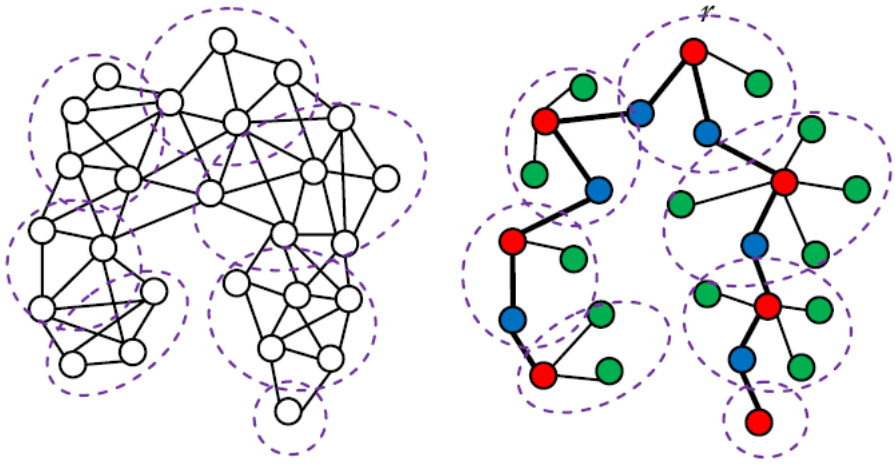
2.1. **The model.** The model of a flat wireless sensor network G in this paper is as follows [5-7,15]:

- Each node in the network has a distinct ID. Initially, nodes do not have any knowledge about the network except for their own IDs which are unique.
- Each node repeats transmission or reception, and performs local computation in synchronized fixed intervals, called *rounds*. In each round, a node can act either as a transmitter or as a receiver.
- Nodes have no collision detection capability.
- Communication between nodes is symmetric. This means that if a node v can receive a message from a node u, then u also can receive a message from v.
- There is a single communication channel in the network.

2.2. **Cluster-based structure.** In this section, we define the cluster-based structure for a dynamic wireless sensor network G. In our clustering, the nodes of G are partitioned into node-disjoined clusters. There is one head node, called a cluster-head, in each cluster. A cluster-head is connected to all other nodes called member nodes. Between two neighboured clusters, there is a gateway node which is a member of one cluster but connects the cluster-head nodes of both clusters. The following defines the construction of the clustering.

Definition 2.1. *Let $G = (V, E)$ be a connected undirected graph with a specified node r, a cluster-based network of G, called as cluster-based network of G and denoted as $CNet(G)$, is a spanning tree of G with root r. In $CNet(G)$, each node knows its status: either as cluster-head, or as gateway, or as pure-member.*

Figure 1 shows an example of the construction of a cluster-based structure $CNet(G)$ (Figure 1(b)) from a flat sensor network G (Figure 1(a)). Here, the red, blue and green coloured nodes (Figure 1(b)) are the cluster-head, gateway and pure-members nodes, respectively, in $CNet(G)$.



(a) Flat sensor network G (b) Cluster-based network $CNet(G)$

FIGURE 1. Example of a cluster-based wireless sensor network

The structure of $CNet(G)$ is defined recursively as follows:

If G contains one node r, then r is the root of $CNet(G)$ and r is a cluster-head.

Let $G = (V, E)$ and $CNet(G) = (V, E_{CNet(G)})$ be its cluster-based network. Let $G' = (V \cup \{new\}, E \cup E')$ be a graph obtained by adding a node *new* to G, where $E' = \{(new, u) | u \in V\}$ and *new* and u are in the transmission range with each other. Then cluster-based network of G' is defined as $CNet'(G) = (V \cup \{new\}, E_{CNet} \cup \{(new, w)\})$, where w is the parent of *new* in $CNet'(G)$. Let U be the set of the nodes in V connected to *new*.

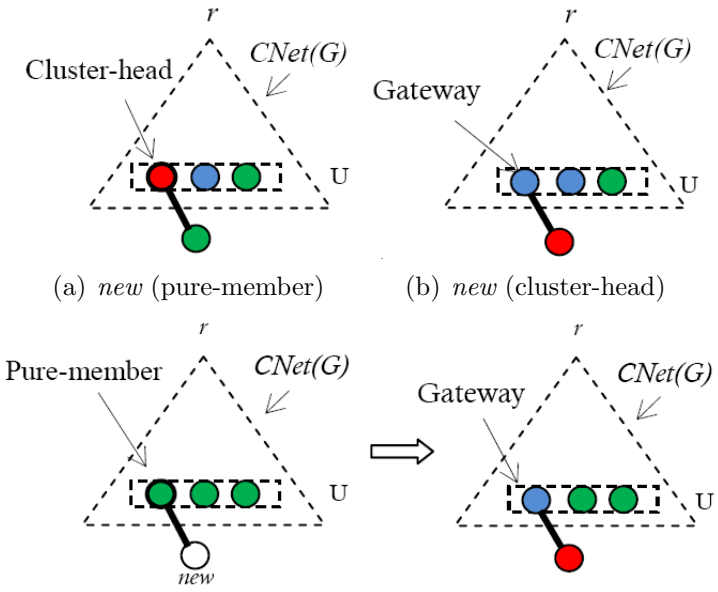
In G', other than *new* and w, the nodes have the same status as they have in $CNet(G)$. The status of w and *new* in $CNet'(G)$ is decided as follows:

If there exist cluster-head(s) in U, select one as w and *new* is a pure-member of w.

Else if there exist gateway(s) in U, select one as w and *new* is a cluster-head (of a new cluster).

Else, U contains only pure-members. Select one as w; then set w to be a gateway and *new* to be a cluster-head (of a new cluster).

Figure 2 shows an example of how the status of a new node in $CNet(G)$ is determined.



(a) *new* (pure-member) (b) *new* (cluster-head)

(c) *new* (cluster-head) and it's parent node (gateway)

FIGURE 2. Status determination process of a new node in cluster-based structure $CNet(G)$

Definition 2.2. *Given a graph G and its cluster-based network $CNet(G)$, a backbone of $CNet(G)$, denoted as $BT(G)$, is a sub-tree of $CNet(G)$ formed by cluster-heads and gateways. $BT(G)$ has the same root as $CNet(G)$ (In Figure 1(b) the thick lines refer to the edges that belong to $BT(G)$, whereas the slim lines are edges in $CNet(G)$).*

Below are some of the properties of the structure.

Property 2.1. *Given a graph G and its cluster-based network $CNet(G)$ where p is the smallest number of complete sub-graphs in G, then $CNet(G)$ and $BT(G)$ have the following properties [5]: (1) $CNet(G)$ has at most p clusters and $BT(G)$ has at most $2p-1$ nodes, (2) there is no edge between cluster-heads in G.*

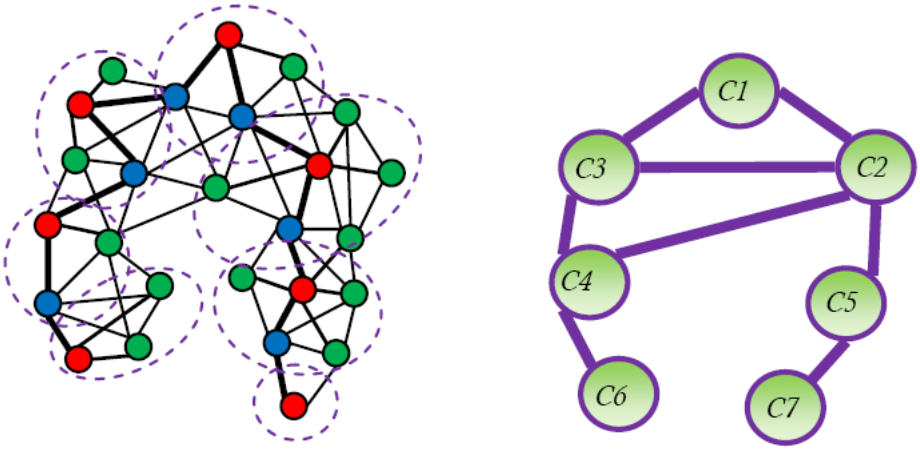
Property 2.2. *Given a graph G with $CNet(G)$ as its cluster-based network, and let $MDS(G)$ be the maximum dominating set of G, then [5]: each node $u \in MDS(G)$ has at most 5 cluster-heads in its neighbors.*

Property 2.3. *Let G be a graph, and $CNet(G)$ be its cluster-based network, then [8]: for each node in $CNet(G)$ of graph G the number of cluster-heads within two hops is less than 20.*

In [5], *Eulerian* $BT(G)$ is described where the message travels an *Eulerian* tour in $BT(G)$ if every undirected edge in $BT(G)$ is replaced by two edges with opposite directions. In the tour, a message called token starts from the source node, visits every node and returns to the source node. At the beginning, the token is in the source node. It then visits each node in H starting from the source node in depth-first order. When a node v receives the token, it sends the token and its ID with the message to one of its neighbors which has not received the token yet. If v has no such neighbor which has not been visited by the token yet, v returns the token to the node from which it received the token for the first time. The movement of the token forms an Eulerian cycle of $BT(G)$. It patrols every node in $BT(G)$ and returns to the source node finally.

Definition 2.3. *Given a graph G with $CNet(G)$ as its cluster-based network, $CGraph(G)$ is a cluster-graph obtained from $CNet(G)$ where each cluster-node in the graph represents a cluster in $CNet(G)$. In the cluster-graph, there exists a cluster-edge between two cluster-nodes, for example, $C1$ and $C2$ if there exists at least a node $u \in V(G)$ or nodes $u, v \in V(G)$ connecting two clusters of $CNet(G)$.*

Figure 3 shows an example of the construction of cluster-graph $CGraph(G)$ from a cluster-based network $CNet(G)$.



(a) Cluster-based Sensor Network $CNet(G)$

(b) Cluster-graph $CGraph(G)$

FIGURE 3. Construction of cluster-graph from $CNet(G)$

We define the node-move-in and node-move-out operations for our dynamic cluster-based network as follows.

Definition 2.4. *Given a graph G and a cluster-based network of G $CNet(G)$, a node-move-in operation is a process where a node new moves into the existing $CNet(G)$ and the network is re-organized to $CNet'(G)$, where G' is the graph obtained by adding new to G'.*

Definition 2.5. *Given a graph G and a cluster-based network $CNet(G)$, a node-move-out operation is a process where a node lev leaves from the existing $CNet(G)$ and the network is re-organized to $CNet'(G)$, where G' is the graph obtained 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 pure-member
- IDs and status of all 1-hop neighbours in G
- neighbouring cluster-heads' IDs in G
- parent's ID (if the node is not the root)
- children's ID and the shortest way to the neighboring clusters (if the node is a cluster-head)

3. Proposed Routing Protocol.

In this section, we present an efficient routing protocol. Unlike the routing protocol proposed in [6] that established a path on the cluster-based structure, this protocol establishes a path from a source node s to a destination

node d on graph G. Nodes that lie in between s and d maintain a routing table which consists only of the destination node's ID and the next hop node towards d.

The protocol performs two basic functions: route discovery and route maintenance. The route discovery procedure is invoked when a source node s wishes to find a fresh route to a destination node d. On the other hand, the route maintenance procedure is called when an intermediate node u does not find the next hop node, say v towards d. For route maintenance, we assume that u does not find v due to its leave from the network and that the reconstruction of $CNet(G)$ has already been completed.

3.1. **Route discovery procedure.** We describe our proposed routing protocol SPRP_{G} below.

Phase 1: s first generates a message *find-cluster-route* then appends its own ID and destination node d's ID to the message.

Phase 2: If s is not a cluster-head, it generates a message *find-route* then appends the source and destination nodes IDs to the message. The message is then sent to its cluster-head.

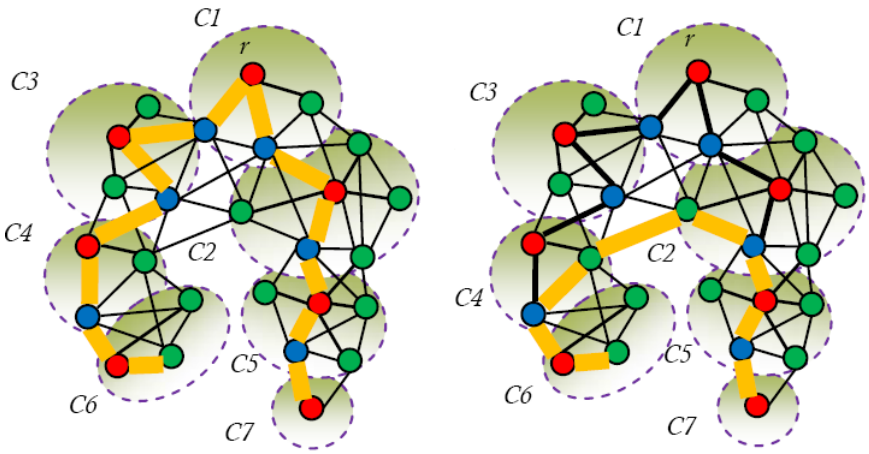
Phase 3: The cluster-head h then generates another message *cluster-info* and performs $Eulerian(BT(G))$ to collect each cluster's neighbors' information in order to form the cluster-graph.

Phase 4: Once the cluster-graph is formed, h computes the shortest (cluster) route to the destination node's cluster using the Breadth-First Search (BFS) technique.

Phase 5: Hereafter, the source cluster-head transmits *find-route* message along with the computed route to the destination cluster-head using the route in the BFS technique. Upon receiving *find-route* message each cluster-head forwards the message to the next cluster-head until the destination node is found.

Phase 6: Once the destination node is found the node found it generates a *route-found* message and sends it to the source node.

Figure 4 shows an example that compares what could be achieved with previous proposed protocols in [5-7] and what could be achieved with our proposed protocol.



(a) Routing approach proposed previously (b) Our proposed routing approach

FIGURE 4. Comparison of routing approaches

3.2. **Route maintenance procedure.** Let u be a node which does not find the next hop node towards d.

Phase 1: u generates a token that contains a *search-route* message then appends d's ID and sends the message back to the source node s.

Phase 2: s calls the *Route Discovery Procedure.*

Theorem 3.1. *Let G be a graph, $CNet(G)$ and its cluster-based network, $CGraph(G)$ be the cluster-graph. Then $SPRP_G$ can be established from a source node s to some destination node d in $O(p)$ rounds, where, p is the number of clusters in $CNet(G)$.*

Proof: In the *Route Discovery Procedure*, first $Eulerian(BT(G))$ is called to collect the cluster information with which to form the cluster graph which requires $O(p)$ rounds. Then the cluster-graph is calculated locally. Once the cluster-graph is developed, the source cluster uses the BFS technique to find the destination cluster node. Finally, the computed cluster-graph route is used to find the destination cluster. The total time requires in this process is $O(p)$ rounds.

Therefore, the total time required in the *Route Discovery Procedure* is $O(p)$.

In the *Route Maintenance Procedure*, the node fails to find its next hop towards the source node which requires $O(p)$ rounds. Then the source node initiates the *Route Discovery Procedure* which requires $O(p)$ rounds. Thus, the total time required in the *Route Maintenance Procedure* is $O(p)$.

Hence, the $\mathrm{SPRP_G}$ protocol requires $O(p)$ rounds to establish the route from the source node to the destination node.

4. **Maintenance of Cluster-based Structure $CNet(G)$.** According to the definition of $CNet(G)$ in Section 2 and the algorithms $\mathrm{SPRP_G}$ in Section 3, each node in $CNet(G)$ is required to have the following knowledge:

(I) It knows its neighbors 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).

(II) Each cluster-head maintains its neighboring clusters' information in graph G. Each cluster-head also maintains a potential member node(s) to send messages to the neighboring cluster(s). For one neighboring cluster only one potential node is selected.

In [5,7] two operations, *node-move-in* and *node-move-out*, are used for constructing and reconfiguring $CNet(G)$, where the nodes of $CNet(G)$ maintain knowledge (I) only. Note that, in [6] each cluster-head maintains its r (> 1) hop members.

In this section, we show how we maintain knowledge (II). There are two ways to construct a $CNet(G)$: one way is to add nodes of G one by one into $CNet(G)$ by using node-move-in operation; and the other way is to do a gossip on G once so that every node knows the knowledge of whole network G which requires $O(n)$ rounds, and then construct the same $CNet(G)$ at each node.

4.1. **Node-Move-In algorithm.** Let graph $G = (V, E)$ have n nodes and $CNet(G) = (V, E_{CNet(G)})$. A graph obtained by adding a node *new* into G' is a graph $G = (V \cup \{new\}, E \cup E')$, where $E' = \{(new, u) | u \in V$, and new and u are in each others' transmission range$\}$. The cluster-based network of G is defined to be $CNet(G) = (V \cup \{new\}, E_{CNet} \cup \{(new, w)\})$, where w is the parent of new in $CNet(G)$. According to [5], a node-move-in operation can be done in $O(d)$ expected rounds, where d is *new's* neighbors in G. Each node in G has knowledge (I) when the operation is finished.

We add two additional phases after the node-move-in operation of [5] as follows:

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

If *new* finds the cluster-head in its neighbor, *new* informs it directly;

Else if there are gateway(s) in its neighbors that are connected to cluster-head(s), *new* chooses the gateway node with lowest ID;

Else *new* chooses the pure-member with the lowest ID for each neighboring cluster that is connected to cluster-head.

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

If the status of *new* is a cluster-head, then h chooses the node that forwarded it the message as the route to cluster h;

Else if *new* is a pure-member, then h updates its route to its neighboring cluster if it finds that *new* is the better option to reach the cluster(s).

Theorem 4.1. *Let $CNet(G)$ be a cluster-based network of G. Then when joined with new into $CNet(G)$ it is expected to take $O(d)$ rounds, where d is the number of neighbors of the new node new of the cluster-based network.*

Proof: According to the Theorem 4 of [5], it requires an expected $O(d)$ rounds to collect neighbouring nodes' information and to determine the status and parent node of *new* , i.e., knowledge (I). In order to achieve knowledge (II) we use Property 2.3. According to the property in the cluster-based structure a node can have at most 19 cluster-heads within its 2-hop neighbours. Thus *new* requires 19 more rounds to inform the neighbouring clusters' information to those cluster-heads. Then to update the information on potential nodes, it requires at most 19 more rounds. Thus the whole processes here can be done in $O(1)$ rounds.

Hence, the total time required for Node-Move-In algorithm is an expected $O(d)$ rounds.

4.2. **Node-Move-Out algorithm.** Let graph $G = (V, E)$ have n $(n \geq 1)$ nodes and $CNet(G) = (V, E_{CNet})$. The graph obtained by deleting a node lev from G is a graph $G = (V - \{lev\}, E - E')$, where $E' = \{(lev, u)|(lev, u) \in E\}$. We assume that the graph G is connected and after the leave the resulting graph is also connected.

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

Assume that C_i $(i = 1, 2, 3, \ldots)$ are the sub-trees of lev in T. Since G is connected, after lev is executed there exists at least one edge e in G which is neither an edge of T nor an edge of G but connects node u of T with node v of H. In [5], $CNet(G)$ with knowledge

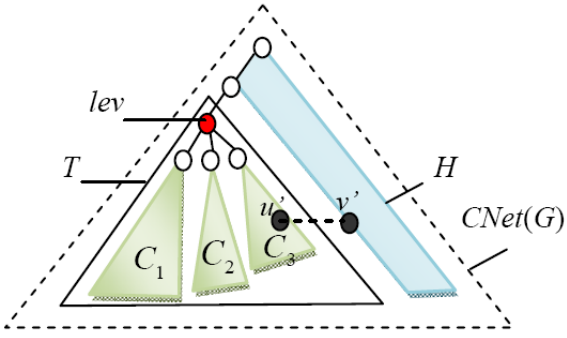


FIGURE 5. $CNet(G)$ is divided into two subtrees T and H

(I) is reconfigured in $O(|T|)$ rounds by using the following two phases:

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

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

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

Phase 3: Finally, once the new clustering is formed, node u' calls *Eulerian*(T'), where u' is the node that found the edge with node v' in H and T' is the subtree rooted by u excluding *lev*. In this procedure each node in T' updates their neighboring 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 neighbor lists and recalculate their neighboring clusters and potential nodes' information.

Theorem 4.2. *Let $CNet(G)$ be a cluster-based network of G, then leave of* lev *from $CNet(G)$ can be done in $O(|T|)$ rounds, where T is the sub-tree rooted by the leaving node* lev.

Proof: Using our Node-Move-In algorithm and Theorem 5 of [5] we can prove the Theorem. In the first phase, node *lev* calls *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 $O(|T|)$ rounds. In the second phase, *Eulerian*(T'), where $T' = T\text{-}\{lev\}$, is called by which nodes in T other than *lev* moves in H determine their status and parents which also take $O(|T|)$ rounds. Finally, in the third phase, *Eulerian*(T') is called by which nodes in T' update their information on neighboring clusters and select potential nodes to the neighboring cluster. Since according to the Property 2.3 each node has at most 19 cluster- heads i.e., clusters in its neighbours according to the node-move-in algorithm this process requires $O(|T|)$ rounds.

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

5. **Simulation.** This section describes the simulation result. The experiments are carried out on random unit disk graphs that are generated in 1000m $\times$ 1000m square fields. The transmission range of each node is set to 50m. The number n of nodes used in the experiment varies from 200 to 1000. For each number n of nodes the experiments are repeated 10 times, each time by generating a random unit disk graph. The average results are then presented here.

Figure 6 presents a comparison between the lengths of path found in our protocol and in [6]. It is observed that our protocol always finds a path with smaller shorter length than that found in the previous routing protocol.

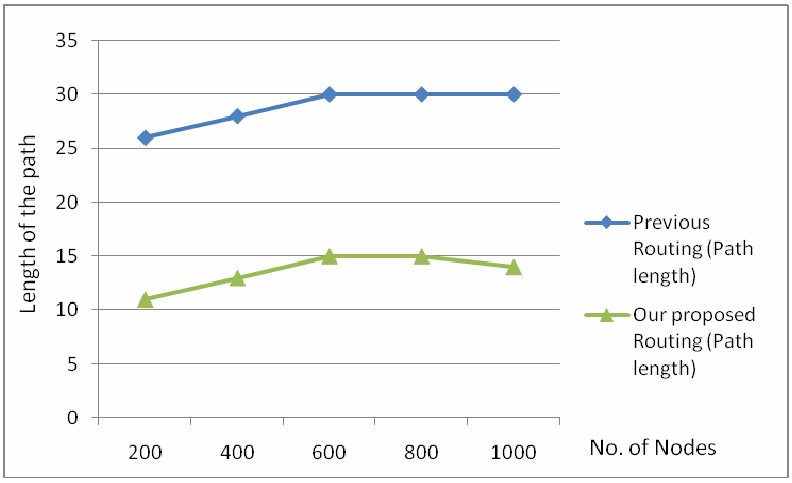


FIGURE 6. Comparison between lengths of path found in our protocol and in [6]

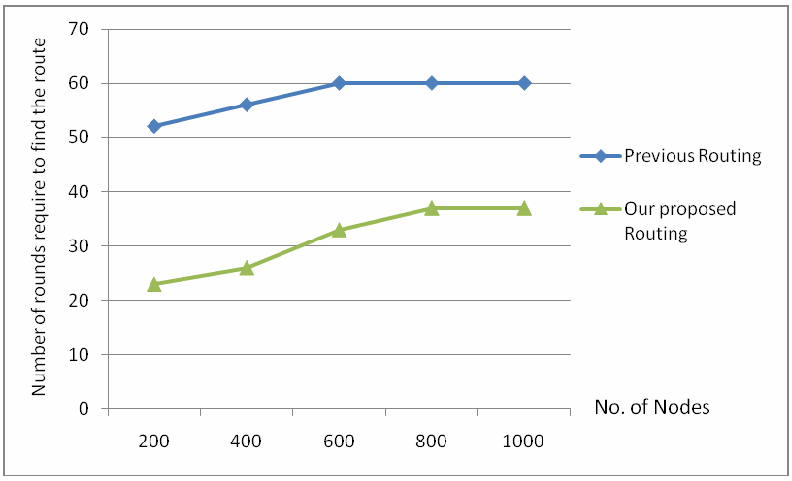


FIGURE 7. Computational complexities on finding the routes

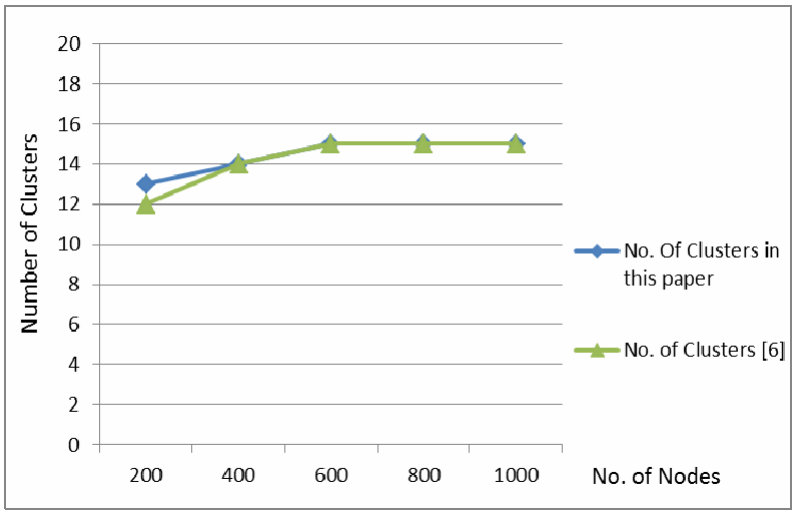


FIGURE 8. Number of clusters in the cluster-based network

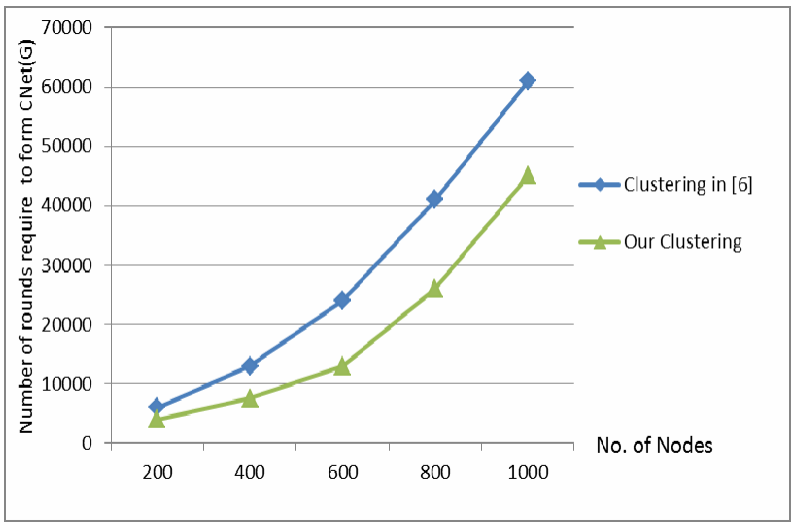


FIGURE 9. Computational complexities on forming cluster-based network

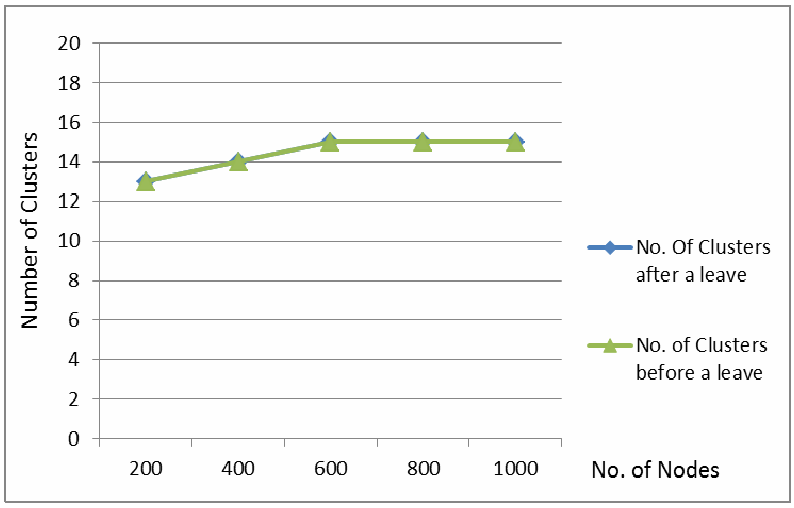


FIGURE 10. Number of cluster after leaving a node from the network

On the other hand, Figure 7 shows the computational complexities required in finding the routes. Here we show a comparison between computational time to find a route using our protocol and in the protocol presented in [6]. It is observed that our routing protocol uses less computational time. This is because once the destination node is found the route is then created using the path on graph G, compared with the route on the cluster-based structure which requires the same path to be travelled.

Figure 8 shows the cluster information in the network after n number of nodes form a cluster-based network. Initially an increased number of clusters can be seen when the number of nodes in the network is small. However, the network grows larger the number of clusters decreases proportionally, thus showing the effectiveness of the proposed model. On the other hand, Figure 9 shows the computational complexity of forming the network. Here we show a comparison between computational time to form a cluster-based network using the proposed model and the technique presented in [6,7]. It is observed that our proposed model uses less computational time. This is because in the proposed cluster-based model nodes have to maintain less status information than in [6] and once a node

joins an existing network it does not have to communicate with the root node to update the height of the network.

Finally, in Figure 10, we show a comparison of the effect on the number of clusters between the proposed model and the model presented in [6]. It is observed that in our proposed model once a node leaves an existing network it much less of an effect than in [6].

6. **Conclusion.** In this paper, we have presented an efficient Routing Protocol SPRP_{G} using our improved dynamic cluster-based wireless sensor network. We also have updated *Node-Move-In* and *Node-Move-Out* algorithms to fit into our proposed cluster-based structure. Finally, our experimental results have showed that our proposed protocol finds a better route with less computational time.

In future work, we would like to concentrate on the following aspects. Firstly, would like to design a cluster-based structure with multiple simultaneous node-move-in operations. Here 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.

Moreover, 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 understand that the achievement of fault-tolerance is 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. It is necessary to consider clustering from an arbitrary situation. Finally, 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 close to reality.

Acknowledgment. This work is partially supported by grant GUP Tier 2, 2011 with Reference No. Q.J130000.7143.00J93 and Malaysia-Japan International Institute of Technology (MJIIT) of Universiti Teknologi Malaysia (UTM). The authors gratefully acknowledge the useful comments and suggestions of all reviewers in CrownCom2011 International Conference, Japan, 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. Finally, we would like to thank Mr. Al LaPrade for his help to correct the English language of this paper.

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 fading channel, *Proc. of IEEE SICON*, pp.197-211, 1997.

[3] C. E. Perkins and P. Bhagwat, Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, *Comp. Commun. Rev.*, pp.234-244, 1994.

[4] C. E. Perkins and E. M. Royer, Ad-hoc on-demand distance vector routing, *Proc. of the 2nd IEEE Wksp. Mobile Comp. Sys. and Apps.*, pp.183-97 1999.

[5] J. Uchida, A. K. M. M. 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.

[6] A. K. M. M. Islam, Y. Katayama, W. Chen and W. Wada, A novel cluster-based architecture and routing protocols for dynamic ad-hoc radio networks, *Journal of Electrical Engineering*, vol.EE33, no.I&II, 2006.

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

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

[9] F. G. Cocetti, J. S. Gonzalez and I. Stojmenovic, Connectivity based k-hop clustering in wireless networks, *Telecommunication Systems*, vol.22, no.1-4, pp.205-220, 2003.

[10] S. Basagni, Distributed clustering for ad hoc networks, *Proc. of the International Symposium on Parallel Architectures, Algorithms, and Networks*, pp.310-315, 1999.

[11] D. Dubashi, A. Panconesi, J. Radhakrishnan and A. Srinivasan, Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons, *Proc. of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp.717-724, 2003.

[12] J. Wu and H. Li, On calculating connected dominating set for efficient routing in ad hoc wireless networks, *Telecommunication Systems*, vol.18, no.1-3, pp.13-36, 2001.

[13] D. B. Johnson and D. A. Maltz, Dynamic source routing in ad-hoc wireless networks, in *Mobile Computing*, T. Imielinski and H. Korth (eds.), Kluwer, pp.153-181, 1996.

[14] T. Nakashima and T. Sueyoshi, A performance simulation for stationary end nodes in ad hoc networks, *International Journal of Innovative Computing, Information and Control*, vol.5, no.3, pp.707-716, 2009.

[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*, no.45, pp.104-126, 1992.

[16] C. Wang, T. Hong, G. Horng and W. 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.