An Efficient Routing Protocol on a Dynamic Cluster-based Sensor Network

A.K.M. Muzahidul Islam [1]
Shahrum Shah Abdullah [3]
Malaysia-Japan Int'l Institute of Technology
Universiti Teknologi Malaysia
Kuala Lumpur, Malaysia
[1]akmmislam@ic.utm.my
[3]shahrum@ic.utm.my

Koichi Wada [2]
Department of Computer Science
Nagoya Institute of Technology
Nagoya, Japan
[2]wada@nitech.ac.jp

Jiro Uchida [4]
Software Development Division
AISIN AW Co., Ltd.
Okazaki, Japan
[4]cepter1218@ybb.ne.jp

Wei Chen [5]
Department of Computer Science
Tennessee State University
Nashvilee, USA
[5]wchen@tnstate.edu

Abstract—**This paper presents an efficient routing protocol on a Dynamic Cluster-based Wireless Sensor Network (WSN). The cluster-based architecture presented here is constructed dynamically. Unlike some previous routing protocol that finds the route on cluster-based architecture which is a spanning tree of graph *G*, our proposed protocol finds route on *G*. The cluster-based architecture is dynamic, thus it is self-constructible and reconfigurable and is facilitated by two topology management operations: *node-move-in* and *node-move-out*. For these 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 architecture, it requires *O(p)* rounds, where *p* is the number of clusters in the network. In a scenario where the number of sensor nodes *n* is enormous, *p* is much less than *n*. Finally, our simulation results describe that the proposed routing protocol finds a better route with less length.**

Keywords- Wireless Sensor Network; Routing; Node-Move-in; Node-Move-out; Dynamic Cluster-based Network.

I. 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 to an external base-station. The sensor nodes have a large coverage area and longer range. They have higher degree of fault tolerance than other wireless networks: failure of one or few nodes does not affect the operation of the network. They are also self-configuring or self-organizing [1]. Clustering is used to leverage the underlying flat sensor network topology and provide a hierarchical organization [4-6], [8-10].

The topology of a WSN changes when its physical condition is changed. While the topology and geography of a WSN changes, its network architecture and network functions need to be reorganized. Considering the 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 getting out* of and *nodes joining into* an existing network. Even for stationary nodes, when battery is low, it must get out and go to charge mode. Then, the charged nodes should join back to the network once again. Once a hierarchical clustering established, the maintenance of the cluster organization turns to be crucial in the presence of network topology changing.

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 do so in an efficient manner [2], [3], [5], [11], [12]. To minimize communication overhead and facilitate energy efficient routing we construct a cluster-based architecture to a flat Dynamic WSN, where the maintenance of architecture is done through node-move-in and node-move-out operations [4], [5].

In [4], a novel Cluster-Based Architecture for a Dynamic WSN is presented, where the maximum radius of a cluster one. To perform an efficient broadcasting the architecture then constructs and maintains a Communication Highway called Backbone Tree *(BT)*. Broadcasting is done on this architecture using the size of the BT. However, no routing protocol for this architecture is proposed.

Later, in [5], to find a routing path better than the size of BT, the authors have proposed another novel cluster-based architecture where the maximum radius of clusters is more than one. To perform an 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 has shown that to find a route on graph *G* it requires *O(n)* rounds (where *n* is the number of the

This research is partially supported by grant GUP Tire-2, 2011 and Malaysia-Japan International Institute of Technology, 2011 of Universiti Teknologi Malaysia.

CROWNCOM 2011, June 01-03, Osaka, Japan

DOI 10.4108/icst.crowncom.2011.245879

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.

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

To solve this problem, in this paper, we propose an efficient routing protocol on graph G using our improved cluster-based architecture. We also study maintenance of the cluster-based architecture.

First, we describe the improved cluster-based architecture in Section II. Then in Section III, we present our routing protocol. Here we show that to create a route on graph G using the cluster-based architecture it requires $O(p)$ rounds, where p is the number of clusters. In Section IV, we show the maintenance algorithms for the cluster-based architecture. Finally, in Section V, we present some simulation results.

II. Cluster-Based Architecture for Dynamic Wireless Sensor Networks

In this section, we describe the model and the cluster-based architecture 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 represents the sensor nodes and E is the set of edges. In G, nodes u and v have an edge between them *iff* they are in each others' transmission range.

A. The Model

Below is the model of a flat wireless sensor network [4-6]:

- Initially, each node knows self ID only and it's unique.
- Transmission/reception and local computation are performed in synchronized fixed intervals, called *rounds*. In each round, a node can act either as a transmitter or as a receiver.
- No collision detection is available.
- Communication is symmetric.
- Single communication channel in the network.

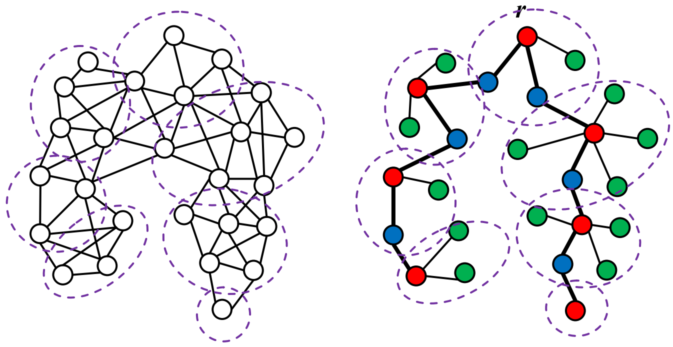
B. Cluster-Based Architecture

In our clustering, the nodes of G are partitioned into node-disjoined clusters. There is one head node in each cluster which connects to all other member nodes. Between two neighboured clusters, there is a gateway node which is the member of one cluster but connects the head nodes of both clusters (Figure.1). The following definition gives the construction of the clustering.

Definition 1 Given a graph $G = (V, E)$ with a specified node r, a cluster-based network of G, called as *cluster-based network* or simply *cluster-net* 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*.

The architecture 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.



a. Flat Sensor Network G b. Cluster-based Network $CNet(G)$

Figure 1: Cluster-based Sensor Network example

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 $E' = \{(new,u) \mid u \in V\}$ 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 $CNet'(G)$ are 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 1 shows an example of construction of a cluster-based architecture $CNet(G)$ (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 head, gateway and pure-members nodes, respectively, in $CNet(G)$.

Definition 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 $CNet(G)$ (Figure. 1(b)) .

Below are some of the properties of the architecture.

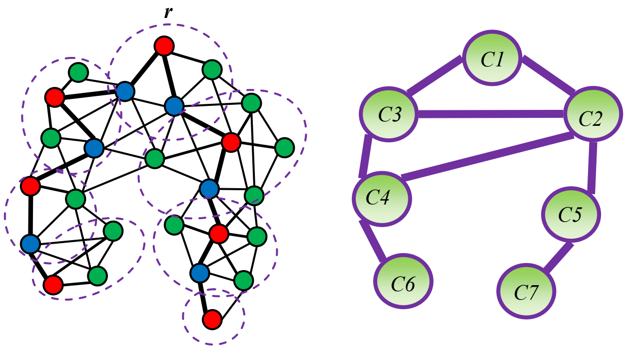
Property 1 Given a graph G and its cluster-based network $CNet(G)$ where p is the smallest number of complete sub-graphs in G, then $CNet(G)$ and $BT(G)$ have the following properties [4]: (1) $CNet(G)$ has at most p clusters and $BT(G)$ has at most $2p$-1 nodes, (2) there is no edge between cluster heads in G.

Property 2 Given a graph G and $CNet(G)$, and let $MDS(G)$ be the maximum dominating set of G, then [4]: each node $u \in MDS(G)$ has at most 5 cluster-heads in its neighbors.

Property 3 Let G be a graph, and $CNet(G)$ be the cluster-based network, then [7]: for each node in $CNet(G)$ of graph G the number of cluster-heads within two hops is less than 20.

In [4-6], *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, each node of $BT(G)$ transmits the message at least once, and exactly one node transmits the messages at each round.

Definition 3 Given a graph G and $CNet(G)$, *CGraph(G)* is a 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 2 shows an example of construction of cluster-graph *CGraph(G)* from a cluster-based sensor network *CNet(G)*.



a. Cluster-based Sensor Network *CNet(G)* b. Cluster-graph *CGraph(G)*

Figure 2. 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 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 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.

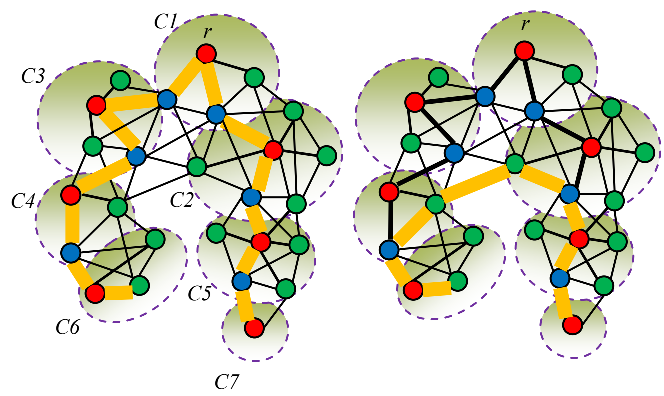
- self ID;
- self status i.e. cluster-head, gateway, or pure-member;
- IDs and status of all 1-hop neighbors in G;
- neighboring cluster-heads' ID in G;
- parent's ID (if the node is not the root);
- children's ID and the shortest way to the neighboring clusters (if the node is a cluster-head);

III. Proposed Routing Protocol $SPRP_G$

In this section, we present an efficient routing protocol. Unlike the routing protocol proposed in [5] that established a path on the cluster-based architecture, 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 only maintain routing table which consists 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 the reconstruction of *CNet(G)* has already been completed.

A. RouteDiscovery Procedure



a. Routing approach proposed previously b. Our proposed Routing approach.

Figure 3. Comparison of Routing approaches.

We describe proposed routing protocol $SPRP_G$ below.

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

Phase 2 if s is not a cluster-head, then 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 to form cluster-graph.

Phase 4 once cluster-graph is formed, h computes the shortest (cluster) route to the destination node's cluster using 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 next cluster-head until the destination node is found.

Phase 6 Once the destination node is found the node that finds the node generates *route-found* and sends the message back to the source node.

Figure 3 shows an example, that gives a comparison of the routing protocols proposed or could be achieved in [4], [5], [6], with our proposed protocol.

B. RouteMaintenance Procedure

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

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

Phase 2 *s* calls *RouteDiscovery Procedure.*

Theorem 1: Let *G* be a graph, *CNet(G)* be the 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 *RouteDiscoveryProcedure*, first *Eulerian*(*BT*(*G*)) is called to collect the cluster information 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 *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 requires in the *RouteDiscovery Procedure* is *O(p).*

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

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

IV. Maintenance of Cluster-based Architecture *CNet*(*G*)

According to the definition of *CNet*(*G*) in Section II and the algorithms for *SPR* in Section III, each node in *CNet*(*G*) needs 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, it potential node(s) to send message to the neighboring cluster.

In [4], [6] 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 [5] 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 another way is to do a gossip on *G* once so that every node knows the knowledge of whole network *G* which needs *O(n)* rounds, and then construct a same *CNet*(*G*) at each node.

A. 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_{old} is a graph $G=(V\cup\{new\}, E\cup E')$, where $E'=\{(new,u) \mid u\in V$ and new and u are in each others' transmission range$\}$. Cluster-net 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 [4], a node-move-in operation can be done in $O(d)$ expected rounds, where d is the neighbors of new in *G*. Each node in *G* has knowledge (I) when the operation finished.

We add two additional phases after the node-move-in operation of [4] 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 to it directly;

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

Else *new* chooses the pure-member with the lowest ID that is connected with the 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 better option to reach the cluster(s).

Theorem 2: Let *CNet(G)* is a cluster-based network of *G*, then joins of *new* into *CNet(G)* can be done in expected $O(d)$ rounds, where *d* is the number of neighbors of the new node *new* of the cluster-based network.

Proof: Using Property 3 and the theorem 4 of [4] we can prove the theorem easily. Due to the space limitation we omit the detail here.

B. Node-Move-Out Algorithm

Let graph $G = (V, E)$ have n ($n \geq 1$) nodes and $CNet(G) = (V, E_{CNet})$. A graph obtained by deleting a node *lev* from G is a graph $G = (V - \{lev\}, E - E')$, where $E' = \{(lev, u) \mid (lev, u) \in E\}$. It is assumed 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*) (the case that *lev* is the root of *CNet*(*G*) can be dealt similarly. Assuming that C_i (i = 1,2,3,…) are the sub-trees of *lev* in *T*. Since *G* is connected, after *lev* leaves, there exits 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 [4] *CNet*(*G*) with knowledge (I) is reconfigured in O(|*T*|) rounds by using the following two phases:

Phase 1 *lev* calls *Eulerian*(*T*) in which a message "find an edge not belonging to *T*" to find the edge *e* = (*u, v*).

Phase 2 node *u* calls *Eulerian*(*T*) in which the message is "move into *H*" to start a *Eulerian* tour in *T* from node *u* until all the nodes of *T* moved into *H*.

However, to maintain knowledge II the following 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 *H* and *T'* is the subtree rooted by *u*. In this procedure each node in *T'* updates their neighboring cluster information as in *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' information.

Theorem 3: 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 subtree rooted by the leaving node *lev*.

Proof: Using our Node-Move-In algorithm and Theorem 5 of [4] we can prove the Theorem easily. Thus, due to the space limitation we omit the detail here.

V. SIMULATION RESULTS

The experiments are carried on random unit disk graphs that are generated in 1000m x 1000m square fields. The transmission range of each node is set to 50m. The number *n* of nodes is used in the experiment varies from 200 to 1000.

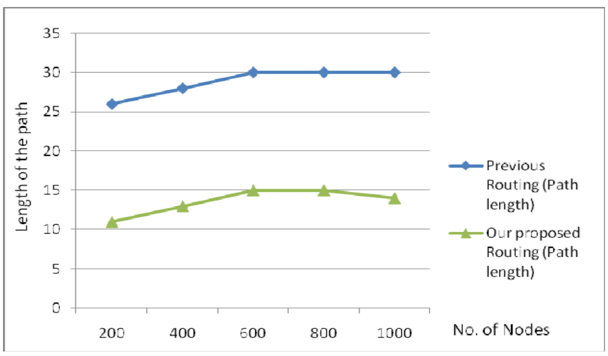


Figure 4. Comparison of path length between our protocol and [5].

For each number of nodes the experiments are repeated 10 times, each time by generating a random unit disk graph. The average results are then presented here. It is observed that our protocol finds a route with smaller length than the length is found in [5].

VI. CONCLUSIONS

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 for the cluster-based architecture. Finally, our experimental results have showed that our proposed protocol finds a better route than that of in [4], [5], [6]. In future work, we would like to establish secured communication architecture for our dynamic cluster-based architecture where nodes could communicate with each other in a secure manner.

REFERENCES

[1] C.-C. Chiang. "Routing in Clustered Multihop, "*Mobile Wireless Networks with Dading Channel*", Proc. IEEE SICON '97, Apr. 1997, pp. 197-211.

[2] C.E. Perkins and P. Bhagwat. "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, " Comp. Commun. Rev.,Oct. 1994, pp.234-44.

[3] C.E. Perkins and E. M. Royer. "*Ad-hoc On-Demand Distance Vector Routing*", Proc. 2nd IEEE Wksp. Mobile Comp. Sys. And Apps., Feb. 1999, pp. 183-97.

[4] 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, Number 1-2, pp. 1-31, 2008.

[5] A.K.M. Muzahidul Islam, Y. Katayama, W. Chen, W. Wada, "A novel cluster-based architecture and routing protocols for dynamic ad-hoc radio networks,"*Journal of Electrical Engineering*, *The Institution of Engineers, Bangladesh*, vol. EE33, No. I & II, 2006.

[6] Wei Chen, A.K.M. M. Islam, Mohan Malkani, Amir Shirkhodaie, Koichi Wada, and Mohamed Zein-Sabatto, "*NovelBroadcast/Multicast Protocols for Dynamic Sensor Networks*", IEEE International Parallel & Distributed Processing Symposium 2007, USA.

[7] M. Goldberg, Packing of 14, 16, 17 and 20 Circles in a Circle, *Mathematics Magazine*,Vol. 44, No. 3 (May, 1971), pages 134–139.

[8] F. Garcia Cocetti, J. Solano Gonzalez, I. Stojmenovic "Connectivity Based k-Hop Clustering in Wireless Networks", Telecommunication Systems 22:1-4, 205-220, 2003.

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

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

[11] J. Wu, 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.

[12] D. B. Johnson and D. A. Maltz. "Dynamic Source Routing in Ad-hoc Wireless Networks", Mobile Computing, T. Imielinski and H. Korth, Eds., Kluwer, 1996, pp. 153-81.