Open access • Proceedings Article • DOI:10.1109/GLOCOM.2010.5683454

Overlapping Clusters Algorithm in Ad Hoc Networks — Source link

Nevin Aydin, Farid Nait-Abdesselam, Volodymyr Pryyma, Damla Turgut

Institutions: Istanbul Arel University, university of lille, University of Central Florida

Published on: 01 Dec 2010 - Global Communications Conference

Topics: Wireless ad hoc network, Mobile ad hoc network, Network topology, Load balancing (computing) and Cluster analysis

Overlapping Clusters Algorithm in Ad hoc Networks

Nevin Aydin
Istanbul Arel University
Dept. of Industrial Engineering
34295 Sefakoy-Istanbul, Turkey
Email: nevinaydin@arel.edu.tr

Farid Naït-Abdesselam
University of Lille
Computer Science Research Laboratory
59655 Villeneuve d'Ascq Cedex - France
Email: farid.nait-abdesselam@lifl.fr

Volodymyr Pryyma and Damla Turgut
University of Central Florida
Dept. of EECS
4000 Central Florida Ave., Orlando 32816
Email: {vpryyma,turgut}@eecs.ucf.edu

***Abstract*—Clustering allows efficient data routing and multi-hop communication among the nodes. In this paper, we propose Overlapping Clusters Algorithm (OCA) for mobile ad hoc networks. The goal of OCA is to achieve network reliability and load balancing. The algorithm consists of two discrete phases. The start-up phase takes battery and bandwidth capacity, transmission range, density, mobility, and buffer occupancy as input parameters to perform initial clustering for the entire network. The maintenance phase monitors the status of the network and keeps the network topology updated through local and global re-clustering. We compare the performance of OCA with Lowest ID, Highest Degree, WCA, and LCC algorithms in YAES simulator. The simulation results show that OCA outperforms all the compared algorithms in terms of network reliability and load distribution. The average numbers of global re-clusterings and reaffiliations were much lower in OCA than the other algorithms. However, OCA generates a larger number of clusters, which is expected considering that the nodes are allowed to be members of multiple clusters at the same time.**

I. Introduction and Motivation

Infrastructure-based networks have a native organization which is based on dedicated nodes playing the role of routers and switches. Ad hoc networks do not have such a native organization; any node can play the role of the router. Clustering algorithms impose a self-organized hierarchical structure on the flat organization of the ad hoc network, and thus contribute to efficient routing.

There is a need to efficiently route data packets among the nodes in an ad hoc network such that communication over multi-hop paths in the network becomes possible. However, it is often difficult to coordinate communication among nearby nodes due to the lack of infrastructure, as opposed to other types of networks, and due to the constant mobility of the nodes. In order to make ad hoc networks more efficient, a hierarchical structure can be introduced, where the entire network is partitioned into segments called clusters.

This paper introduces a novel clustering protocol for wireless ad hoc networks, called the Overlapping Clusters Algorithm (OCA). The proposed algorithm consists of two distinct phases. The start-up phase performs initial partitioning of a network into clusters, while the maintenance phase updates the topology of a network as time goes on. Depending on the situation, OCA performs either local or global re-clustering.

The rest of the paper is organized as follows. Section II summarizes the related work. The description our proposed algorithm is presented in Section III. We present the simulation results in Section IV and Section V concludes the paper.

II. Related Work

Clustering algorithms should have two properties: dominance property and and two-hop property [7]. Clustering algorithms that satisfy the ad hoc clustering properties can easily provide controlled access to the bandwidth and scheduling of the members in each cluster [10]. While not all clustering protocols satisfy both properties, many of the existing algorithm do satisfy all of the above properties.

The Weighted Clustering Algorithm (WCA) [4] is very simple in the way that the clusterheads are elected. The clusterheads are chosen based on the calculated weights of the nodes. DWCA [6], FWCA [8], and WBCA [15] are all extensions of WCA that employ some sort of a weight-based cluster selection. The DMAC [1] and Weighted Highest Degree [13] are also weight-based clustering algorithms.

In the Highest Connectivity clustering algorithm, the highest degree node is always selected as the clusterhead by the adjacent nodes within the same cluster [9]. Similarly, in the Lowest ID algorithm a node with the lowest local id is always chosen as a clusterhead. In order to resolve the frequent clusterhead changes, the Least Cluster Change (LCC) algorithm limits the change of clusterheads to specific conditions [5]. Hybrid Energy-Efficient Distributed (HEED) clustering algorithm is applicable to both ad hoc as well as sensor networks [16]. This particular algorithm takes a probabilistic approach in selecting clusterheads for a sensor network. Trust and Mobility-based clustering algorithm [12] uses a specific trust parameter to create a secure network of clusters, while Fault-Local Self-Stabilizing clustering algorithm [11] creates a network of non-overlapping clusters with approximately the same size. An access-based self-organizing clustering scheme [3] uses a randomized control channel broadcast system to generate stable clusters with minimal maintenance overhead. This scheme maximizes the worst-case control channel efficiency. A clustering scheme with adjustable transmission range is proposed in [14]. This approach is naturally energy-efficient and manages to reduce overall network density.

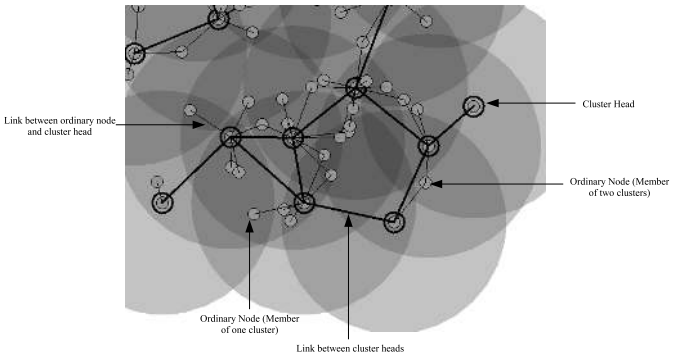

Fig. 1. Screenshot of the cluster formation using OCA in YAES simulator.

III. Overlapping Clusters Algorithm (OCA)

A. Problem definition

Even though the existing clustering algorithms can efficiently divide a network into clusters, only a few of them focus on improving network reliability and load distribution. Due to this imbalance some nodes can expire prematurely. The algorithm presented in this paper, called Overlapping Clusters Algorithm (OCA), sets the network reliability and fair load distribution as its main priorities.

B. Algorithm preliminaries and properties

The OCA is a combination of a Weighted Clustering Algorithm [4] and Least Cluster Change algorithm [5]. However, more focus is applied on maintaining overall network stability. Thus, the clusters are allowed to overlap, and ordinary nodes have the ability to join multiple clusters with different membership level for each cluster.

Figure 1 is a screenshot of the cluster formation using OCA. It clearly shows that some nodes are able to connect with more than one clusterhead. This ability to join multiple clusters is a distinct feature of OCA.

The Overlapping Clusters Algorithm consists of two distinct phases. The first phase deals with initial division of the network into clusters, while the second phase focuses on maintaining and updating the network topology.

Each node in the network is assigned a status indicator, which classifies it either as a clusterhead, an ordinary node, or a neutral node. The significance of the status indicator is discussed later in more detail.

The following are the brief descriptions of the algorithm properties:

Dominance property. Initially all nodes start out as neutral, being neither a clusterhead nor an ordinary node. At the end of initial clustering procedure, every node will either be a clusterhead itself, or it will be a member of at least one cluster. Thus, the proposed algorithm satisfies the dominance property.

Two-Hop Property. Since the dominance property is satisfied, as discussed above, all nodes in the network are either clusterheads or ordinary nodes. Therefore, it takes a node at most two hops to reach another node in the same cluster. So the OCA satisfies the two-hop property.

C. Clusterhead election procedure

The following is a detailed description of each algorithm step:

Start-up phase:

1) Assign parameter values to each node (Please see Table I for more details).
2) Assign the weight values for all input parameters. These weights can be arbitrarily adjusted to make the algorithm more suitable for specific situations, provided that the sum of individual weights adds up to one.
3) Each node is initially in neutral status and calculates its weight value based on the input parameters. The total weight function is used to calculate the weight values for all of the nodes.
4) Every node transmits its weight value to all of the nodes it can reach. After this is done, the nodes compare their own weights with all the other weights that they received.
5) If a node's weight is lower than any other weight it has received, this node stays in the neutral position until all of the higher weighted nodes within its reach make their own decisions of becoming either a clusterhead or an ordinary node. However, if a node has the highest weight value, when compared to all the other weights it has received, then that node becomes a clusterhead.
6) Once it is a neutral node's turn to decide, it first looks to join any existing clusterheads. If a clusterhead is found, the node will attempt to join it, and if it is successful then it will switch its status to that of an ordinary node. If a clusterhead rejected the node, then the node will keep on looking for another one. If more than one clusterhead is found and is able to accept a node, then that node will try to join all of them. However, if no clusterheads are found, then the node will become a clusterhead itself. Once all of the nodes have decided on their status, the network is completely divided into clusters. At this point, the OCA begins its maintenance phase.

Maintenance phase:

1) Every time period, T, the parameters for each node are updated based on the network conditions at the current time period.
2) Immediately after the new parameters have been assigned, each node calculates its new weight value using the same weight function as the first time.
3) Next, each node transmits the newly calculated weight to all the nodes it can reach. The clusterheads compare their weights to the received ones. If an ordinary node in any cluster has a higher weight value than the clusterhead, then that cluster is dissolved and a local re-clustering takes place.
4) If any one of the nodes becomes isolated, or disconnected from all of the existing clusterheads, then a global re-clustering is triggered.
5) If any two of the clusterheads move to within a certain distance of each other, then the algorithm checks for the

number of nodes under their control. If either cluster if heavily loaded, then both are left as they were. However, if both clusters are lightly loaded, then they are dissolved and local re-clustering is performed.

6) Finally, each node monitors the capacity of every clusterhead that it belongs to, in order to evenly distribute the load among the clusters.

Algorithm 1 presents the pseudo code that describes the actual implementation of OCA in YAES simulator.

Algorithm 1 Overlapping Clusters Algorithm

```
init
for i = 0 to 100 do
    move nodes
    update node parameters
    calculate new node weights
    for j = 0 to n do
        check node status
        if status = N then
            init
        else if status = C then
            check clusterhead position
            if next to other clusterhead AND buffer = empty
            AND load <= max/2 then
                local re-clustering
            end if
        else
            if node weight > clusterhead weight then
                local re-clustering
            end if
        end if
    end for
    check load distribution
    update membership status
end for
```

IV. Simulation Study

A. Simulation environment

In order to show that OCA creates a reliable network structure through clustering and provides fair load distribution among clusters, a simulation study was performed in YAES [2] simulator. A typical scenario in which OCA can be implemented is an ad hoc network with a predefined number of mobile nodes enclosed in a finite area.

The simulation area measured is 100 x 100. The number of nodes is varied from $N = 50$ to $N = 200$. The transmission range is also varied from $R = 10$ to $R = 70$ units. In each simulation cycle, the nodes are set to move in random direction by some displacement value. The maximum displacement value is set to 5 units. Table I outlines the specific environment settings for the simulation in more detail.

Each node is assigned a weight value based on certain input parameters. The higher the weight value, the more desirable it is to have that node as a clusterhead. The input parameters for the weight function are battery capacity, transmission range, bandwidth capacity, density of local area around a node, mobility, and buffer occupancy. The total weight equation, used to calculate the weight for any single node is as follows:

TABLE I
Simulation Parameters

Simulation Parameter	Value	Range
area	100x100	
optimal cluster size	10	
transmission range	70	10-70
number of nodes	200	50-200
max node displacement	5	0.5-5
battery capacity weight (w_1)	0.05	
transmission range weight (w_2)	0.05	
bandwidth capacity weight (w_3)	0.1	
density weight (w_4)	0.1	
mobility weight (w_5)	0.3	
buffer occupancy weight (w_6)	0.4	
simulation time units	100	

$$W_n = w_1 \cdot BC_n + w_2 \cdot TR_n + w_3 \cdot BD_n + w_4 \cdot DC_n + w_5 \cdot \frac{1}{MB_n} + w_6 \cdot \frac{1}{BF_n}$$

Battery capacity (BC) reflects the battery state, or remaining power, of a node at any given time. Transmission range (TR) indicates how far a node is able to reach. Bandwidth capacity (BD) means how many nodes can be connected to the node at the same time. Density (DC) gives the number of individual nodes located in the area around any single node. Mobility (MB) shows how fast each node moves around in the network. Finally, the buffer occupancy (BF) tells us how busy a clusterhead is at a given point in time.

Since the weight function considers a reciprocal value of mobility and buffer occupancy, these two parameters have a lower bound so that the weight value of any node does not become infinity.

B. Simulation metrics

In order to analyze the performance of the OCA, the following metrics were observed during the simulation: the average number of clusters, the average number of reaffiliations per unit time, and the average number of global re-clustering per unit time. The results were obtained by varying the transmission range of the nodes and their mobility.

C. Simulation results

The number of clusters formed in a network provides insights into the efficiency and the stability of the network. Ideally, it is desirable to have just the right number of clusters so that the entire network area is covered.

Figure 2 shows a summary of results for the average number of clusters formed versus transmission range. For this particular scenario, the number of nodes is fixed at $N = 100$, and the mobility variable is set to 5. The transmission range, however, is varied in this case. As can be seen in Figure 2, the OCA generates considerably larger number of clusters for low transmission range as oposed to other algorithms used in the

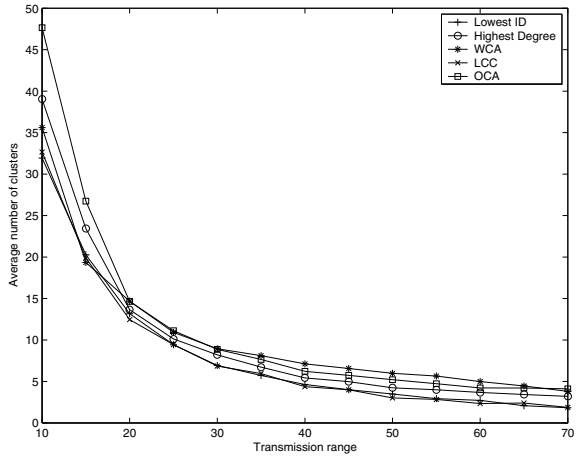

Fig. 2. Average number of clusters vs. transmission range

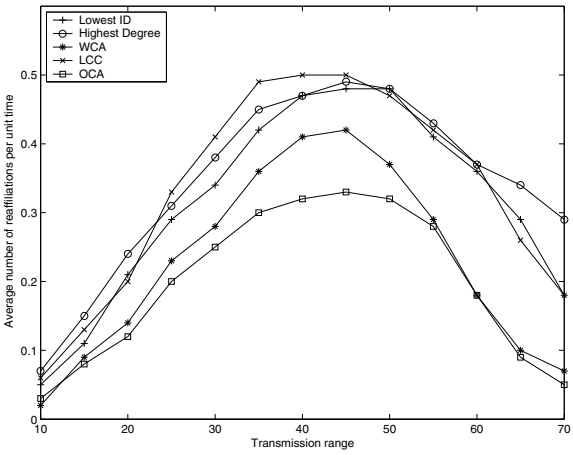

Fig. 4. Average number of reaffiliations vs. transmission range

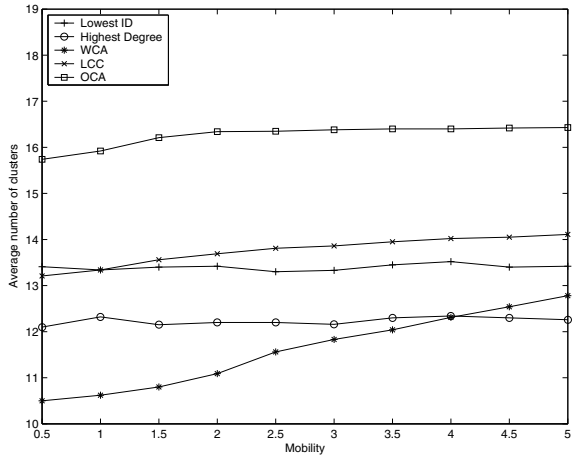

Fig. 3. Average number of clusters vs. mobility

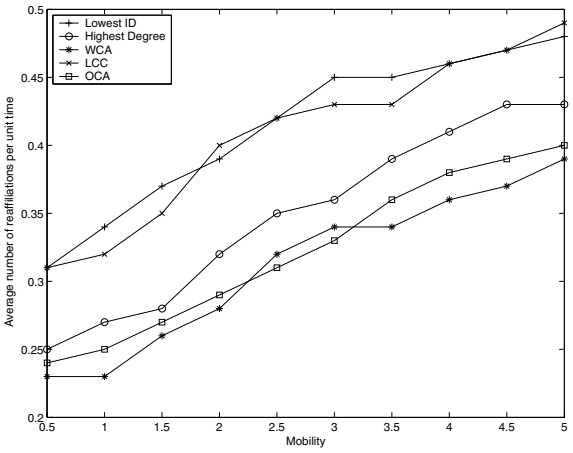

Fig. 5. Average number of reaffiliations vs. mobility

comparison. This is due to the fact that the proposed algorithm allows for overlapping clusters.

A performance comparison for the average number of clusters formed versus mobility is given in Figure 3. For this scenario, the transmission range is fixed at $R = 20$, while the mobility parameter is veried from 0.5 to 5. The number of nodes is fixed at $N = 100$. It is evident from the results of Figure 3 that mobility has only a small effect on the number of clusters formed. The average number of clusters stayed fairly constant for all of the examined algorithms and only increased slightly with increasing mobility.

The next metric, average number of reaffiliations per unit time, provides further insights into the network's stability. Figure 4 displays the simulation results for the average number of reaffiliations versus the transmission range. In this case the mobility variable remains constant, while we vary the transmission range. The OCA appears to have fewer reaffiliations per unit time than the other algorithms used in this comparison. Overall, the number of reaffiliations is lowest when both the number of nodes and the transmission range are either very low or very high.

Figure 5 presents the simulation results for average number of reaffiliations versus mobility. For this particular simulation scenario, the number of nodes is set to $N = 100$ and the transmission range remains constant. There appears to be a linear relationship between the number of reaffiliations and mobility. The OCA has about the same number of reaffiliations per unit time as WCA for this particular metric. Number of isolated nodes encountered indicates the overall reliability of the network. This simulation metric was observed by counting the number of times that the global re-clustering method was incurred per unit time.

Figure 6 shows a summary of the simulation results for the average number of global re-clustering per unit time versus the transmission range. For this simulation scenario, the mobility variable remains constant, while the transmission range is varied. Since the proposed algorithm allows the clusters to overlap, there exist areas in the network that are covered by more than one cluster. This greatly increases the overall network reliability. As can be seen in Figure 6, the number of times when global re-clustering is incurred decreases greatly with increasing transmission range.

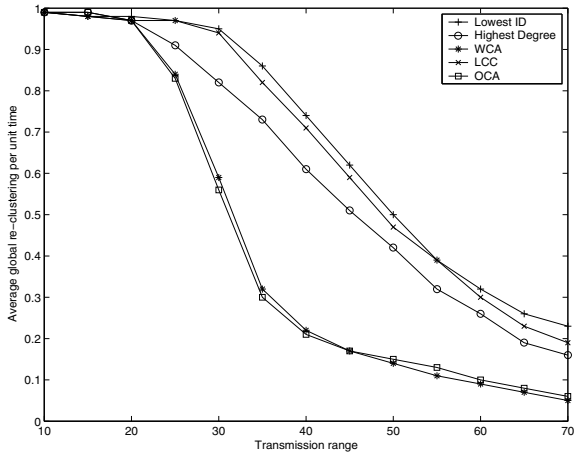

Fig. 6. Average number of global re-clustering vs. transmission range

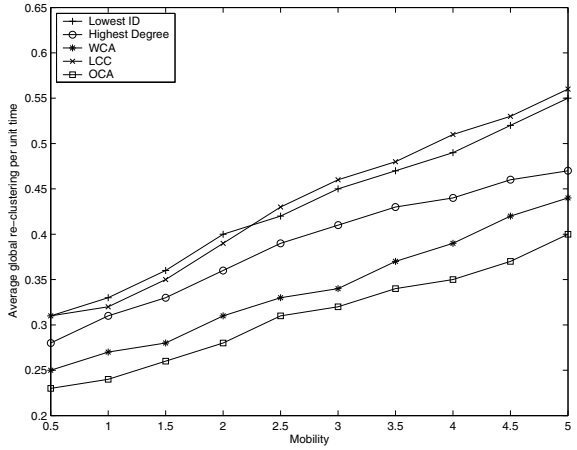

Fig. 7. Average number of global re-clustering vs. mobility

The results for the average number of global re-clustering per unit time versus mobility can be seen in Figure 7. The number of nodes for this simulation is set to $N = 100$, and the transmission range of each node is set to $R = 20$. As with the number of reaffiliations, there appears to be a linear relationship between the number of global re-clustering inccured and mobility. For this particular simulation scenario, OCA appears to perform better by allowing global re-clustering to happen fewer number of times than the other algorithms.

V. Conclusion

This paper proposes a clustering algorithm which is an extension of WCA and implements the limiting conditions for changing clusterheads of the LCC protocol. OCA creates a reliable network of clusters based on the following input parameters: battery capacity, transmission range, bandwidth capacity, density of local area, mobility, and buffer occupancy.

Detailed comparison analyses with Lowest ID, Highest Degree, WCA, and LCC algorithms show that OCA performs better than all of these protocols when it comes to network stability and reliability. The number of global re-clustering and the number of reaffiliations were much lower in OCA than the other algorithms. However, OCA naturally generates a larger number of clusters since it allows individual nodes to become members of multiple clusters. Thus, OCA may be the prefered algorithm in applications that require high reliability but do not focus much on efficient cluster formation.

References

[1] S. Basagni. Distributed Clustering for Ad Hoc Networks. In *Proceedings of the IEEE International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'99)*, page 310, 1999.

[2] L. Bölöni and D. Turgut. YAES - a modular simulator for mobile networks. In *Proceedings of the 8th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'05)*, pages 169–173, October 2005.

[3] Z. Cai, M. Lu, and X. Wang. Channel access-based self-organized clustering in ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(2):102–113, April–June 2003.

[4] M. Chatterjee, S.K. Das, and D. Turgut. WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks. *Journal of Cluster Computing (Special Issue on Mobile Ad hoc Networks)*, 5(2):193–204, April 2002.

[5] C. Chiang, H.-K. Wu, W. Liu, and M. Gerla. Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel. In *Proceedings of 5th IEEE Singapore International Conference On Networks (SICON'97)*, pages 197–211, April 1997.

[6] W. Choi and M. Woo. A distributed weighted clustering algorithm for mobile ad hoc networks. In *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06)*, page 73. IEEE Computer Society, 2006.

[7] C.-R. Dow, J.-H. Lin, S.-F. Hwang, and Y.-W. Wang. An Efficient Distributed Clustering Scheme for Ad-hoc Wireless Networks. *IEICE Transactions on Communications*, E85–B(8):1561–1571, 2002.

[8] Z. El-Bazzal, M. Kadoch, B.L. Agba, F. Gagnon, and M. Bennani. A flexible weight based clustering algorithm in mobile ad hoc networks. In *Proceedings of the IEEE International Conference on Systems and Networks Communication (ICSNC'06)*, page 50, 2006.

[9] M. Gerla and J. Tsai. Multicluster, Mobile, Multimedia Radio Network. *Journal of Wireless Networks*, 1(3):255–265, 1995.

[10] C.R. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks. *IEEE Journal of Selected Areas in Communications*, 15(7):1265–1275, 1997.

[11] V. Mittal and V. Kulathumani. A fault-local self-stabilizing clustering service for wireless ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 17(9):912–922, 2006.

[12] A. Rachedi and A. Benslimane. Trust and mobility-based clustering algorithm for secure mobile ad hoc networks. In *Proceedings of the International Conference on Systems and Networks Communication (ICSNC'06)*, page 72, 2006.

[13] H. Taniguchi, M. Inoue, T. Masuzawa, and H. Fujiwara. Clustering Algorithms in Ad-hoc Networks. *IEICE Transactons on Information and Systems*, J84–D1(2):127–135, 2001.

[14] J. Wu and F. Dai. Virtual backbone construction in MANETs using adjustable transmission ranges. *IEEE Transactions on Mobile Computing*, 5(9):1188–1200, September 2006.

[15] W.-D. Yang and G.-Z. Zhang. A weight-based clustering algorithm for mobile ad hoc network. In *Proceedings of the Third International Conference on Wireless and Mobile Communications (ICWMC'07)*, page 3. IEEE Computer Society, 2007.

[16] O. Younis and S. Fahmy. HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad-hoc Sensor Networks. *IEEE Transactions on Mobile Computing*, 3(4):366–379, 2004.