A Load-Balancing and Energy-Aware Clustering Algorithm in Wireless Ad-Hoc Networks

Wang Jin, Shu Lei, Jinsung Cho, Young-Koo Lee,
Sungyoung Lee*, and Yonil Zhong

Department of Computer Engineering, Kyung Hee University, Korea
{wangjin, sl8132, sylee, zhungs}@oslab.khu.ac.kr
{chojs, yklee}@khu.ac.kr

Abstract. Wireless ad-hoc network is a collection of wireless mobile nodes dynamically forming a temporary communication network without the use of any existing infrastructure or centralized administration. It is characterized by both highly dynamic network topology and limited energy. So, the efficiency of MANET depends not only on its control protocol, but also on its topology and energy management. Clustering strategy can improve the performance of flexibility and scalability in the network. With the aid of graph theory, genetic algorithm and simulated annealing hybrid optimization algorithm, this paper proposes a new clustering strategy to perform topology management and energy conservation. Performance comparison is made between the original algorithms and our two new algorithms, namely an improved weighting clustering algorithm and a novel Genetic Annealing based Clustering Algorithm (GACA), in the aspects of average cluster number, topology stability, load-balancing and network lifetime. The experimental results show that our clustering algorithms have a better performance on average.

1 Introduction

Wireless ad hoc wireless network is a collection of wireless mobile nodes that self-configure to form a network without the aid of any established infrastructure [1]. It can be rapidly deployed and reconfigured where the communication infrastructure is either unavailable or destroyed. However, it is confronted with many challenges too, such as the mobility of hosts, the dynamic topology, the multi-hop nature in transmission, the limited bandwidth and battery, etc. So, the study of MANET (Mobile Ad-hoc NETwork) is a very demanding and challenging task.

Up to now, there are many routing protocols based on various strategies in MANET, and they can be classified into several kinds as follows: (1) proactive and reactive; (2) flat and hierarchical; (3) GPS assisted and non-GPS assisted, etc. These kinds of protocols can be used solely or together. Here we mainly discuss the hierarchical routing protocols, which are based on the clustering algorithm [2, 3].

The rest of the paper is organized as follows. In section 2, some relevant background and commonly used clustering algorithms are presented. Based on which, an

* Corresponding author.

T. Enokido et al. (Eds.): EUC Workshops 2005, LNCS 3823, pp. 1108 – 1117, 2005.

improved clustering algorithm is proposed in section3. In section 4, another novel Genetic Annealing based Clustering Algorithm (GACA) is given so as to optimize the overall network performance. The simulation results and comparison is made in the aspects of average cluster number, topology stability, load-balancing and network lifetime in section 5. Section 6 concludes the paper.

2 Related Work

Similar to the cellular network, the MANET can be divided into several clusters. Each cluster is composed of one clusterhead and many normal nodes, and all the clusterheads form an entire dominant set. The clusterhead is in charge of collecting information (signaling, message, etc.) and allocating resources within its cluster and communicating with other clusterheads. And the normal nodes communicate with each other through their clusterhead, no matter they are in the same cluster or not.

Several original clustering algorithms have been proposed in MANET. These include: (1) Highest-Degree Algorithm; (2) Lowest-ID Algorithm; (3) Node-weight Algorithm; (4) Weighted Clustering Algorithm. (5) Others, like RCC (Random Competition based Clustering), LCC (Least Cluster Change), LEACH etc. We will give some of them a brief description as follows.

2.1 Highest-Degree Algorithm

The Highest-Degree Algorithm was originally proposed by Gerla. and Parekh [4,5]. A node x is considered to be a neighbor of another node y if x lies within the transmission range of y. The node with maximum number of neighbors (i.e., maximum degree) is chosen as a clusterhead.

Experiments demonstrate that the system has a low rate of clusterhead change but the throughput is low under the Highest-Degree Algorithm. As the number of nodes in a cluster increases, the throughput drops and hence a gradual degradation in the system performance is caused. All these drawbacks occur because this approach does not have any restriction on the upper bound of node degree in a cluster.

2.2 Lowest-ID Algorithm

This Lowest-ID Algorithm was originally proposed by Baker and Ephremides [6]. It assigns a unique id to each node and chooses the node with the minimum id as a clusterhead.

As for this algorithm, the system performance is better compared with the Highest-Degree Algorithm in terms of throughput. But it does not attempt to balance the load uniformly across all the nodes.

2.3 Node-Weight Algorithm

Basagni et al. [7] proposed two algorithms, namely distributed clustering algorithm (DCA) and distributed mobility adaptive clustering algorithm (DMAC). In these two approaches, each node is assigned a weight based on its suitability of being a clusterhead. A node is chosen to be a clusterhead if its weight is higher than any of its neighbor's weight; otherwise, it joins a neighboring clusterhead.

Results show that the number of updates required is smaller than the Highest-Degree and Lowest-ID Algorithms. Since node weights vary in each simulation cycle, computing the clusterheads becomes very expensive and there are no optimizations on the system parameters such as throughput and power control.

2.4 Weighted Clustering Algorithm

The Weighted Clustering Algorithm (WCA) was originally proposed by M. Chatterjee et al.[8]. It takes four factors into consideration and makes the selection of clusterhead and maintenance of cluster more reasonable. As is shown in equation (1), the four factors are node degree difference, distance summation to all its neighboring nodes, velocity and remaining battery power respectively. And their corresponding weights are w_1 to w_4 .Besides, it converts the clustering problem into an optimization problem and an objective function is formulated.

$$W_i = w_1\Delta_i + w_2 D_i + w_3 V_i + w_4 E_i \tag{1}$$

However, only those nodes whose neighbor number is less than a fixed threshold value can be selected as a clusterhead in WCA. It is not very desirable in the practical application. For example, many well-connected nodes whose neighbor number is larger than the fixed threshold might be a good candidate as well. Besides, its energy model is too simple. It treats the clusterhead and the normal nodes equally and its remaining power is a linear function of time, which is also not very desirable. So, we proposed an improved clustering algorithm as follows.

3 The Improved Weighted Clustering Algorithm

From the discussion mentioned above, we can see that most clustering algorithms, except for the WCA, only take one of the following factors into consideration, such as the node degree, ID, speed or remaining power. When the problem in one aspect is solved, some other problems are introduced simultaneously. Inspired by the basic idea of WCA, we proposed an improved clustering algorithm.

On the one hand, WCA only chooses those nodes whose neighbor number is less than a fixed threshold as a clusterhead candidate. However, many well-connected nodes whose neighbor number is larger than the fixed threshold might be a good candidate as well. So we can also treat them as clusterheads candidates and select an affordable number of normal nodes from their neighboring nodes. On the other hand, we established a more practical energy-consumption model which we will explain later.

By solving the optimization problem of min (W_i), the clusterheads and their affiliated normal nodes are selected and a trade-off is made from four aspects.

3.1 Principles of the Improved Weighted Clustering Algorithm

In order to determine the fitness value W_i of a node as a clusterhead, we need to consider from the following four aspects.

If the node degree is higher, then the node is more stable as a clusterhead. Here we make a simple conversion $\Delta_i = |N_i - M|$, where N_i is the practical degree of node i and M is the maximum degree. The smaller Δ_i is, the better node i will be as a clusterhead. As for those nodes whose practical degree is larger than the maximum degree M, we also treat them as clusterhead candidates. Once they are chosen as clusterheads, we will choose M nodes with less W_i as their normal nodes. It is a distinctive difference between the original WCA and our improved algorithm, and it can work very well under densely deployed ad hoc networks where the WCA becomes useless.

If the node velocity V_i is lower, then the node will be more stable as a clusterhead.

If the distance summation of node i to all its neighbors D_i is smaller, it will consume less transmission power to communication with the normal nodes within its cluster. In other words, the cost will be smaller.

If the remaining battery power E_i is higher, the longer it will be for node i to serve as a clusterhead. Here we make another conversion and set an energy-consuming model. All the E_i s are set to zero initiatorily. If the node serves as a clusterhead, we assume that it consumes 0.1 unit of energy and if normal node, 0.02 unit of energy. Once some E_i is above 1 (normalized), we believe that this node is out of energy and the network will become useless rapidly due to the avalanche effect [9]. The energy-consuming relationship of 5:1 is commonly used among some papers. And it meets with the minimization problem very well. As for some specific application, one can infer to the related technical report, such as the Mica2 Motes [10].And the model is also applicable through minor modification.

3.2 Steps of the Proposed Algorithm

Taking node i as an example, we compute its W_i according to the following steps and then judge whether it is a clusterhead or a normal node.

Step 1: Compute its practical degree and then derive the equation $\Delta_i = |N_i - M|$.

Step 2: Compute the distance summation D_i to its neighboring nodes.

Step 3: Set the velocity V_i according to the random waypoint mobility model.

Step 4: At first, set E_i to zero and increase their values according to the energy-consuming model. Our algorithm terminates once some E_i is above 1 (normalized).

Step 5: Compute W_i according to various w_i under different application.

Step 6: Taking the node with minimum W_i as the first clusterhead and its neighboring nodes as its normal nodes within the same cluster. Then we go on with this process until all nodes act as either clusterheads or normal nodes.

Step 7: All the nodes move randomly after some unit time and it goes back to step 1 again. And it terminates until a maximum number of time is reached or some node is out of energy.

4 A Novel Genetic Annealing Based Clustering Algorithm (GACA)

The selection of clusterheads set, which is also called dominant set in Graphic Theory, is a NP-hard problem. Therefore, it is very difficult to find a global optimum. So, we can take a further step to use the computational intelligence methods, such as Genetic Algorithm (GA) or Simulated Annealing (SA), to optimize the objective function.

Considering the length of our paper, we will skip the principles of GA and SA, and explain the steps of our new Genetic Annealing based Clustering Algorithm (GACA) directly.

The steps of our GACA are as follows. And it usually takes 5 to 10 iterations to convergence. So, we can say that it converges very fast.

Step 1: As for N nodes, randomly generate L integer arrangements in the range of [1, N].

Step 2: According to these random arrangements and the clustering principle of WCA, derive L sets of clusterheads and compute their corresponding $\sum w_{iold}$.

Step 3: According to the Roulette Wheel Selection and Elitism in GA, select L sets of clusterheads which are better, and replace the original ones.

Step 4: As for each of the L sets of clusterheads, perform the crossover operator and derive the new L sets of clusterheads and their $\sum w_{inew}$.

Step 5: According to the Metropolis "accept or reject" criteria in SA, decide whether to take the one from L sets of clusterheads in $\sum w_{iold}$ or in $\sum w_{inew}$. And the new L sets of clusterheads in the next generation are obtained.

Step 6: Repeat Step 3 to 5 until it converges or a certain number of iteration is reached. And in our simulation, it usually takes 5 to 10 iterations to converge. Then the global optimal or sub-optimal solution min ($\sum w_{inew}$) (i=1, 2…L) is obtained and their corresponding set of clusterheads is known.

In Step 2, we make L random arrangements in order to reduce the randomness in the clustering process, because there is much difference in the set of clusterheads (or dominant set) for different nodes arrangements.

As for the Roulette Wheel Selection in Step 3, we do not take the traditional selection probability $P_i = \frac{\sum w_i}{\sum_{i=1}^{L} (\sum w_i)}$, but $P_i = \frac{e^{-\sum w_i}}{\sum_{i=1}^{L} (e^{-\sum w_i})}$. In that case, the set of

clusterheads whose $\sum w_i$ is smaller will have more chance to be selected. Besides, to overcome the randomness in the probability problem, we preserve the best set of $\sum w_{iold}$ directly to the $\sum w_{inew}$ in Elitism.

To further reduce the randomness and increase the probability that the global solution may occur, we perform M pairs of crossovers as for each of L random arranged integers (i.e. mobile nodes). And the new L sets of clusterheads and their $\sum w_{inew}$ (i=1,2,…L) are derived.

In Step 5, we make the "accept or reject" decision according to the Metropolis criteria. If $\sum w_{inew} \le \sum w_{iold}$, then we accept $\sum w_{inew}$ directly. If $\sum w_{inew} > \sum w_{iold}$, we do not reject it directly, but accept it with some probability. In other words, if $e^{\frac{\sum w_{inew} - \sum iold}{\alpha T}}$ is larger than a randomly generated number in the range of (0,1), which shows that $\sum w_{inew}$ and $\sum w_{iold}$ may be very close to each other, we will still take it. Or else, we will reject the one in $\sum w_{inew}$ and take its counterpart in $\sum w_{iold}$. Besides, we let $T = \alpha T$ (α is a constant between 0 and 1 and we normally take 0.9) after each iteration, so that $\sum w_{inew}$ and $\sum w_{iold}$ must be closer if $\sum w_{inew}$ is to be accepted. In this way, our GACA will not be trapped in the local optima and the premature effect can be avoided. In other words, the diversity of searching space can be ensured and it is similar to the mutation operator in GA.

5 Performance Evaluation

We set our simulation environment as follows. There are N nodes randomly placed within a range of 100 by 100 m^2, whose transmission range varies from 15m to 50m. A Random Waypoint mobility model is adopted here. And our GACA parameters are listed in table 1.

Table 1. GACA parameters

M	L	α	ε
1	10	0.9	0.01

5.1 Analysis of Average Cluster Number

As is shown in figure 1, we simulate N nodes whose transmission range varies from 15m to 50m. We can conclude that:

(1) The average cluster number (ACN) decreases as the transmission range increases.
(2) As for a smaller transmission range, the average number of cluster differs greatly for various N. But when the transmission range is about 50m, one node can almost cover the entire network. So it only takes 3 to 5 clusters to cover all the N nodes.

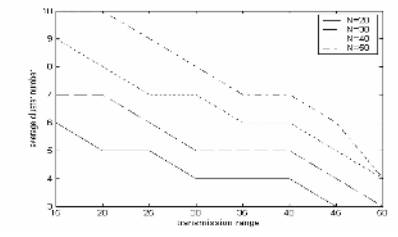


Fig. 1. ACN under various transmission range

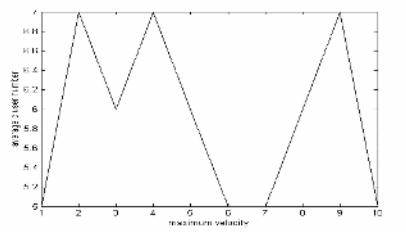


Fig. 2. ACN under various maximum velocities

Besides, we do the same research under various velocities. Taking N=R=30 as an example, we can draw the conclusion from figure 2 that: the average number of cluster varies randomly between 5 and 7, and it is not related with the velocity. In fact, it matches with the practical situation too. For example, when one node with large velocity moves out of a cluster, it is highly possible that some other node gets into the same cluster. Or some of the nodes might move toward the same direction, which results in a relatively slow velocity and a stable cluster too.

5.2 Analysis of Topology Stability

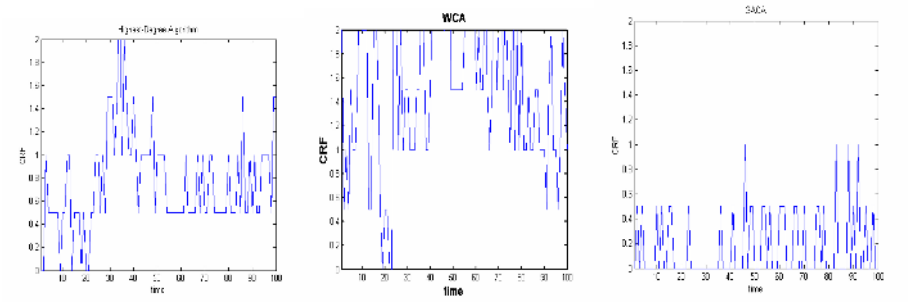
As is mentioned before, the clusterhead and their affiliated normal nodes may change their roles as they move. Here, we define a cluster reaffiliation factor (CRF) as follows:

$$\mathrm{CRF}=\frac{1}{2}\sum_i \left|N_{i1}-N_{i2}\right| \tag{2}$$

here, i is the average number of cluster, and N_{i1}, N_{i2} are the degree of node i at different times. For example, we assume that clusterhead 1 and 2 have 6 and 5 neighbors at first, i.e. $N_{11}=6, N_{21}=5$. As they move after one unit time, their neighbors (degrees) become 5 and 6, i.e. $N_{12}=5, N_{22}=6$. We can derive that CRF is equal to 1. So, we believe equivalently that one node in cluster 1 moves into cluster 2 and one reaffiliation is made.

Under the maximum velocity of 10 m/s, we compared the CRF performance of Highest-Degree Algorithm, WCA and our GACA. From figure 3, we can see that GACA has the lowest CRF, which shows that it is the stablest clustering strategy among three of them. And WCA has the highest CRF value. The average CRF values of them are 1.56, 0.77 and 0.17 respectively.

Besides, we did some other experiments about CRF. We got the conclusion that the CRF increases as the velocity increases.



(a) Highest-Degree Algorithm (b) WCA (c) GACA

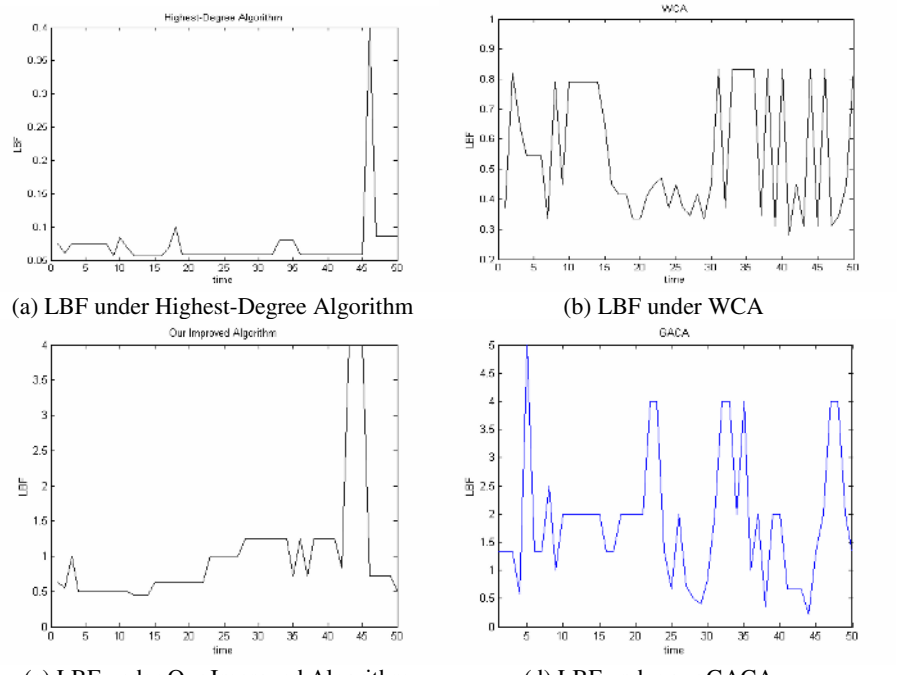
Fig. 3. CRF under various clustering algorithms

5.3 Analysis of Clusterhead Load-Balancing

We take the same definition of load-balancing factor (LBF) as is defined in [8]:

$$\text{LBF}=\frac{n_c}{\sum_i (x_i-\mu)^2}, \quad \mu=\frac{(N-n_c)}{n_c}$$

where, n_c is the average cluster number, N is the number of all nodes, and x_i is the practical degree of node i. The larger LBF is, the better the load is balanced. Taking



(a) LBF under Highest-Degree Algorithm (b) LBF under WCA

(c) LBF under Our Improved Algorithm (d) LBF under our GACA

Fig. 4. LBF under various clustering algorithms

N=20, M=4 as an example. The ideal case is that there are 4 clusters and each clusterhead has a degree of 4, i.e. $n_c = x_i = 4$. Then, $\mu = (20-4)/4 = 4$. So LBF is infinite, which shows that the load is perfectly balanced.

For simplicity, we do not consider the factor of network lifetime here (we will discuss it later in next section). So we set the simulation parameters as follows. (X,Y)=[100,100], N=20, R=30, M=4, maximum velocity $V_{\max} = 5$ and $w_1 = 0.7, w_2 = 0.2, w_3 = 0.1, w_4 = 0$. It should be noted that we make N_i as our primary focus of attention $(w_1 = 0.7)$, because it represents the matching degree of the practical case and ideal case directly. Figure 4 shows the LBF distribution under Highest-Degree Algorithm, WCA, our improved weighted clustering algorithm and GACA. From figure 4 we can see that: the Highest-Degree Algorithm has the worst performance, WCA is secondary to it, and our two improved clustering algorithms are better. Besides, the WCA will become useless under densely deployed ad hoc networks while our algorithm still works well. And their average values are 0.09, 0.38, 1.19 and 1.86 respectively.

5.4 Analysis of Network Lifetime

Finally, we made a comparison between the aforementioned four clustering algorithms in the aspect of network lifetime, as is shown in figure 5. From which, we can see that GACA achieves the best performance, our improved weighted clustering algorithm is second to it, the WCA and the Highest-Degree algorithm are worse.

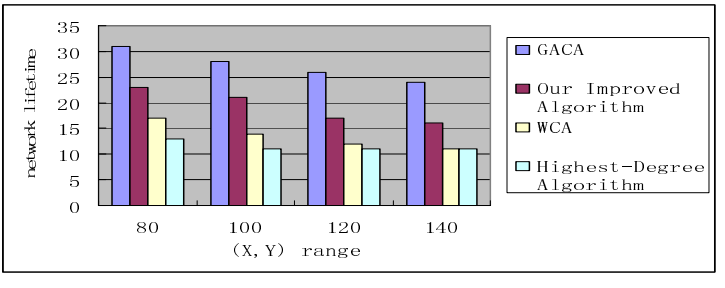


Fig. 5. Network lifetime under various clustering algorithms

6 Conclusion

We proposed an improved weighted clustering algorithm based on the WCA and another novel Genetic Annealing based Clustering Algorithm (GACA) in this paper. Some performance comparison is made in the aspect of average cluster number, topology stability, load-balancing and network lifetime. The simulation results show that our two clustering algorithms have a better performance on average.

Acknowledgement

This work was supported by grant No. R01-2005-000-10267-0 from Korea Science and Engineering Foundation in Ministry of Science and Technology.

References

1. Internet Engineering Task Force MANET Working Group. Mobile Ad Hoc Network (MANET) Charter [EB/OL]. Available at http://www.ietf.org/html.charters/manet-charter.html.
2. C.C. Chiang. Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel [C]. Proceedings of IEEE SICON'97, April1997, pp.197-211.
3. Mingliang Jiang, Jinyang Li, Y.C. Tay. Cluster Based Routing Protocol [EB/OL]. August, 1999 IETF Draft.
4. A.K. Parekh. Selecting routers in ad-hoc wireless networks [C]. Proceedings of the SBT/IEEE International Telecommunications Symposium, August 1994.
5. M. Gerla and J.T.C. Tsai. Multicluster, mobile, multimedia radio network [J]. ACM/Baltzer Wireless Networks, 1(3),1995, pp. 255-265.
6. D.J. Baker and A. Ephremides. The architectural organization of a mobile radio network via a distributed algorithm [J]. IEEE Transactions on Commicationuns COM-29 11(1981) pp. 1694–1701.
7. S. Basagni. Distributed clustering for ad hoc networks [C]. Proceedings of International Symposium on Parallel Architecture*s*, Algorithms and Networks, June 1999, pp. 310–315.
8. M. Chatterjee, S.K. Das and D. Turgut. An On-Demand Weighted Clustering Algorithm (WCA) for Ad hoc Networks [C]. Proceedings of IEEE GLOBECOM 2000, San Francisco, November 2000, pp.1697-1701.
9. Nishant Gupta, Samir R. Das. Energy-aware On-demand Routing for Mobile Ad Hoc Network. [EB/OL], Available form http://crewman.uta.edu/~choi/energy.pdf.
10. MICA2 Mote Datasheet, http://www.xbow.com/Products/Product_pdf_files/ Wirelesspdf/ 6020-0042-01_A_MICA2.pdf, 2004.