A Probabilistic Clustering Algorithm in Wireless Sensor Networks

Hesong Huang and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431
Email: hhuang3@fau.edu, jie@cse.fau.edu

***Abstract*— A wireless sensor network consists of nodes that can communicate with each other via wireless links. One way to support efficient communication between sensors is to organize the network into several groups, called clusters, with each cluster electing one node as the head of cluster. The paper describes a constant time clustering algorithm that can be applied on wireless sensor networks. This approach is an extension to the Younis and Fahmy method [1]. The simulation results show that the extension can generate a small number of cluster heads in relatively few rounds, especially in sparse networks.**

keywords: Wireless sensor networks, cluster head, probabilistic algorithms.

I. Introduction

In a wireless sensor network, all nodes are alike. There is no base station to coordinate the activities of nodes. Thus, all nodes have to make decisions individually. All communications are over wireless links. A wireless link can be established between a pair of nodes only if they are within the wireless transmission range of each other. The wireless links are symmetric: a node can send the message to another node if and only if it can receive from the other end of link.

Sensor networks are widely used in military surveillance, building security, and harsh physical environments. Inexpensive sensor nodes are deployed to the sensing area with little mobility and high density. The sensor nodes have very limited battery power and computing capability. Hence, the sensor networks should be well organized to meet the task. The objective of the clustering algorithm is to partition the network into several clusters. The advantages of clustering algorithm are:

- reducing routing table size,
- reducing the redundancy of exchanged messages,
- reducing the energy consumption, and
- extending the networks lifetime.

Younis and Fahmy proposed HEED (Hybrid Energy-Efficient Distributed clustering) [2] as a probabilistic clustering algorithm. This paper will propose an improved version of HEED. The extension includes three phases:

1) perform the core algorithm [2] to eliminate some nodes,
2) apply the original HEED algorithm, and
3) apply the core algorithm again instead of simply electing all uncovered nodes as cluster heads.

The rest of this paper is organized as follows: Section 2 describes existing various clustering formation approaches. Section 3 presents the new approach of cluster head (CH) election. Section 4 presents the simulation results. Section 5 concludes this paper.

II. Related Work

Many clustering algorithms for ad hoc and sensor networks were proposed in the last few years. The traditional clustering algorithm [1] (also called the general clustering algorithm) is as follows: each node sends an election message including node ID or cost to each of its neighbors, and receives the information from its neighbors. Each node checks if there are some CHs around. If the CHs exist, it will keep silent; otherwise, it will elect itself as CH. The time complexity of traditional clustering is O(n) in the worst case and O($\log n$) on average. The message complexity is O(1) for one node, and O(n) for the networks.

In [2], a clustering algorithm called HEED (Hybrid Energy-Efficient Distributed clustering) is proposed. The pseudo-code of the HEED algorithm executed by each node is given below:

```
I. Initialize
1.   S_nbr ← {v: v lies within my transmission range}
2.   Broadcast S_nbr
3.   is_final_CH ← FALSE

II. Repeat
1.   If ((S_CH ←{v: v is a CH}) ≠ ∅)
2.     my_cluster_head ← least_cost(S_CH)
3.     If (my_cluster_head = NodeID)
4.       If (CH_prob=1)
5.         Cluster_head_msg(NodeID, final_CH, cost)
6.         is_final_CH ← TRUE
7.       Else
8.         Cluster_head_msg(NodeID, tentative_CH, cost)
9.   Else If (CH_prob=1)
10.      Cluster_head_msg(NodeID, final_CH, cost)
11.      is_final_CH ← TRUE
12.  Else If Random(0,1) ≤ CH_prob
13.      Cluster_head_msg(NodeID, tentative_CH, cost)
14.  CH_previous ← CH_prob
15.  CH_prob ← min(2 × CH_prob, 1)
```

16. UNTIL $CH_{provious}$ = 1

III. Finalize
1. If(is_final_CH = FALSE)
2. If (($S_{CH} \leftarrow v$: v is a final CH) $\neq \emptyset$)
3. my_cluster_head $\leftarrow$ least_cost(S_{CH})
4. join_cluster(cluster_head_ID, NodeID)
5. Else Cluster_head_msg(NodeID, final_CH, cost)
6. Else Cluster_head_msg(NodeID, final_CH, cost)

The above algorithm is an iterative process. Initially, each node has a small probability CH_{prob} of becoming a CH. This probability is also used to limit the number of iterations N_{iter}. During each iteration i ($1 \leq i \leq N_{iter}$), every uncovered node (i.e., it has not heard from tentative_CH or a final_CH) volunteers to become a CH with a probability CH_{prob}. Each node maintains a set of neighboring tentative CHs, S_{CH}. A node v_i selects its CH (my_cluster_head) to be the node with the lowest cost in S_{CH}. Note that S_{CH} may include v_i itself if it is selected as a tentative CH. The probability CH_{prob} is doubled after each iteration.

If a node becomes a CH, it broadcasts a message cluster_head_msg(Node_ID, selection_status, cost) to its neighbors, in which the selection_status is set to tentative_CH if $CH_{prob} < 1$, or final_CH if $CH_{prob} \geq 1$.

III. The Proposed Clustering Algorithm

Assume a set of sensors is deployed in a rectangular area:

- Nodes in the area are uniformly distributed.
- The node location is unknown, but every node knows which others are its neighbors.
- The transmission ranges are uniform.
- The link between two nodes is bi-directional.

In the first round the core algorithm [6] is used, where each node will check if its cost is the least among its neighbors (include itself). If it is the node with the lowest cost, it will set itself as core head; otherwise, it will set the least cost neighbor as core.

After the core election, the cluster head election will exclude the non-core nodes. As in HEED, every node has a small initial probability CH_{prob}, which is computed from a predetermined number of iterations. During each iteration, each uncovered node becomes a tentative CH with a probability of CH_{prob}, and resolves conflictions among neighboring tentative CHs with a certain node priority, such as residual energy level and node ID. All nodes double their CH_{prob} for each iteration. The entire election process (step II in the pseudo code) repeats until the values for CH_{prob} of all nodes reach 1.

In the final round, final CHs are considered as CHs, and tentative CHs non-CHs. Every non-CHs must have one neighboring final CH with the highest priority as its CH; otherwise, it is uncovered. All uncovered nodes run the core algorithm to elect some extra CHs. Each uncovered node selects a node with the highest priority in its neighborhood (including itself) as a CH to cover itself.

The pseudo code of the extended algorithm (called the Extended HEED) are as follows:

I. Initialize
1. $S_{nbr} \leftarrow \{v$: v lies within my cluster range$\}$
2. Broadcast S_{nbr}
3. is_final_CH $\leftarrow$ FALSE
4. Get the node ID of the least cost node among its neighbors
5. set the node as my core

II. Repeat
(Exactly same as original algorithm)
UNTIL $CH_{provious}$ = 1

III. Finalize
1. If(is_final_CH = FALSE)
2. If (($S_{CH} \leftarrow v$: v is a final CH) $\neq \emptyset$)
3. my_cluster_heade $\leftarrow$ least_cost(S_{CH})
4. join_cluster(cluster_head_ID, NodeID)
5. Else
6. find the least cost node among its neighbors as CH
7. If (cluster_head_ID = NodeID)
8. Cluster_head_msg(NodeID, final_CH, cost)
9. Else
10. join_cluster(cluster_head_ID, NodeID)
11. Else Cluster_head_msg(NodeID, final_CH, cost)

Figure 1 shows an example of the Extended HEED. There are 500 nodes deployed in a 100 $\times$ 100 area, with a transmission range of 10 and an initial CH_{prob} of 0.01. The algorithm takes 7 rounds and elects 64 CHs (represented by black nodes). A solid line represents a wireless link connecting a non-CH (represented by white nodes) and its corresponding CH. A dotted line represents a link between a non-CH to a CH of a neighboring cluster. In some rare occasions, two CHs are neighbors, which is also represented by a solid line.

IV. Simulation Results

The general clustering algorithm (GC), HEED and Extended HEED (Extended) have been simulated in random networks with 1000 nodes deployed in a 1000×1000 area.

Figure 2 shows the number of CHs of different algorithms, when the transmission range varies from 25 to 400. Under all transmission ranges, GC has the smallest number of the CHs, but GC has the non-constant number of rounds. In the worst case, the number of rounds equals to the network diameter. Both HEED and Extended use a constant number of rounds. Extended has a smaller number of CHs than HEED.

Figures 3 to 5 compare HEED with Extended under different transmission ranges (25, 50, and 100) and numbers of rounds. Simulation results show that Extended elects fewer CHs than HEED under all situations. The difference is more significant with relatively small transmission ranges and small numbers of rounds. In summary, compared with HEED, Extend performs much better in relatively sparse networks, when the number of iterations is limited.

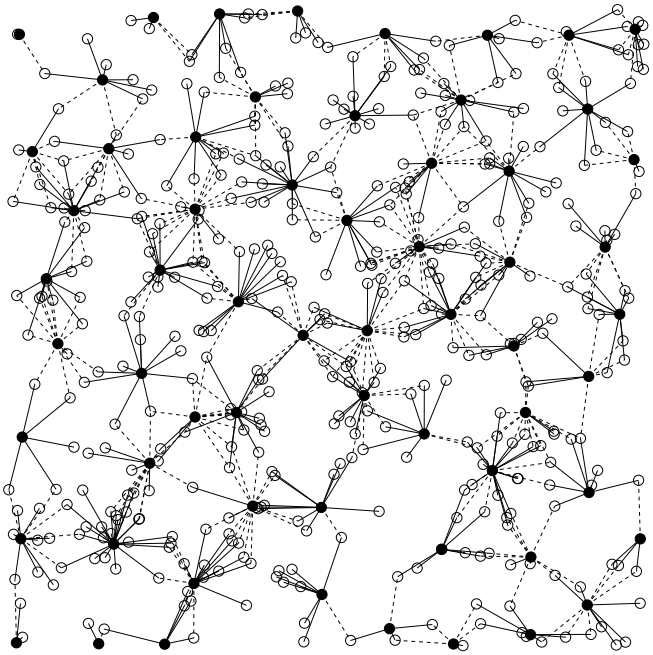

Fig. 1. The result of Extended HEED.

V. Conclusion

In this paper, we presented an extended probabilistic algorithm for Hybrid Energy-Efficient Distributed clustering (HEED) [2]. Because our algorithm adds two more steps to eliminate a large quantity of nodes, and only potential candidates can survive to participate in the cluster head election, it is more efficient than the original HEED. Simulation results have confirmed the above statement.

VI. Acknowledgement

This work was supported in part by NSF grants CCR 0329741, CNS 0434533, CNS 0422762, and EIA 0130806.

References

[1] C. R. Lin, and M. Gerla. Adaptive Clustering for Mobile Networks. In *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, Sep. 1997, pp. 1265-1275.

[2] O. Younis and S. Fahmy. Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach. In *Proceedings of IEEE INFOCOM*, March 2004.

[3] J. S. Liu and C. R. Lin. Power-Efficient Clustering Method with Power-Limit Constraint for Sensor Networks. In *Proceedings of the 2003 IEEE International*.

[4] S. Basagni. Distributed clustering for ad hoc networks. In *Proc. 1999 Int'l Symp. Parallel Architectures, Algorithms, and Networks*, pages 310 -315.

[5] V. Kawadia and P. R. Kumar. Kumar, Power Control and Clustering in Ad Hoc Networks. In *Proc. Infocom 2003.*

[6] P. Sinha, R. Sivakumar and V. Bharghavan. CEDAR: a Core-Extraction Distributed Ad hoc Routing Algorithm. In *INFOCOMM 1999*, pages 202-209.

[7] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. An Application-Specific Protocol Architecture for Wireless Microsensor Networks In *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pages 660-670, October 2002.

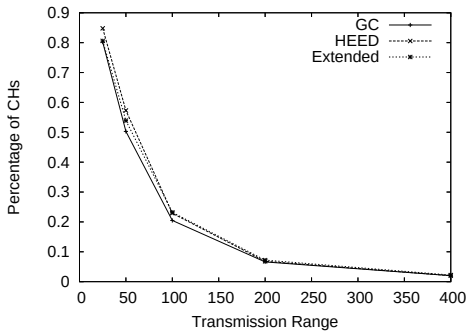

Fig. 2. Number of CHs vs. transmission range.

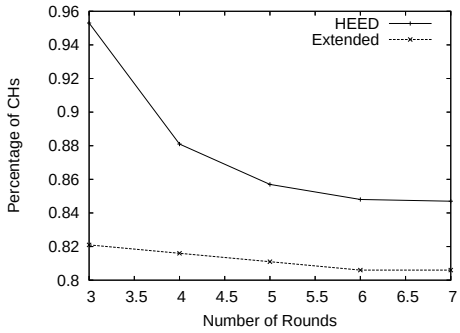

Fig. 3. Number of CHs vs. number of rounds (transmission range is 25).

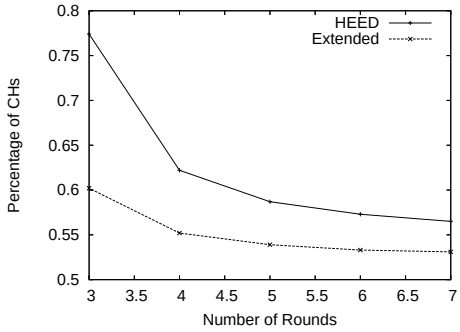

Fig. 4. Number of CHs vs. number of rounds (transmission range is 50).

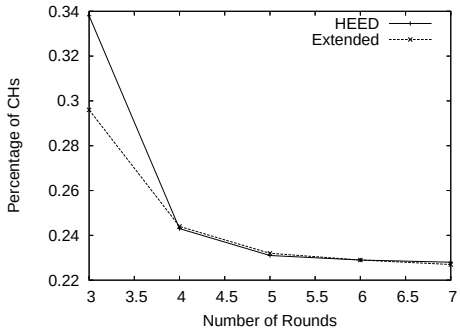

Fig. 5. Number of CHs vs. number of rounds (transmission range is 100).