Energy-Efficient Clustering Algorithm in Wireless Sensor Networks[⋆]

DaeHwan Kim[1,2], SangHak Lee[1] and We Duke Cho[2]

[1] Intelligent IT System Center, Korea Electronics Technology Institute, #68 Yatap-dong, Bundang-gu, Seongnam-si, Gyeonggi-do 463-816, South Korea, {kimdh, shlee}@keti.re.kr

[2] Department of Electrical and Computer Engineering, Ajou University, Suwon, 443-749, South Korea, chowd@ajou.ac.kr

Abstract. Wireless sensor networks is a key technology for new ways of interaction between computers and the physical environment. However, the energy constrained and limited computing resources of the sensor nodes present major challenges in gathering data. Since sensor nodes are densely deployed, redundant data may occur. While cluster-based data gathering is efficient at energy and bandwidth, it's difficult to cluster network efficiently. In this work, a new distributed clustering algorithm for ubiquitous sensor network is presented. Clustering is based on the distance between nodes and the number in a cluster for wireless sensor networks. Simulation results show that the proposed algorithm balances the energy dissipation over the whole network thus increase the amount of data delivery to the sink.

1 Introduction

A sensor network is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it [1]. The number of sensor nodes in a sensor networks can be several orders of magnitude higher than in an ad hoc network [2]. This factor contributes to a drastic increase in traffic of sensor network protocols [3]. Thus it may cause explosive traffic without in-network data processing, e.g. data aggregation. The effective strategy to solve above described problem is cluster-based routing and data aggregation [4]. However, the algorithmic complexity of clustering is known to be NP-hard for finding c optimal clusters in network while it is efficient at energy and bandwidth. Thus, it is difficult to find optimal solution, and consequently clustering is usually performed by some heuristics. The problem we address is to cluster the network distributedly to balance the energy consumption over the whole network while minimizing the overhead of clustering, thus to increase the energy efficiency.

[⋆] This research is supported in part by the Ubiquitous Computing and Network (UCN) Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea

To achieve this goal, we analyze the difference in energy consumption between nodes according to node proximity to its neighbors and propose a new distributed clustering algorithm.

The rest of the paper is organized as follows: in section 2, related works, we survey the previous works related to clustering network. In section 3, we determine the cause of difference in energy consumption between nodes, then get the criteria. Section 4 defines the problem and presents the proposed clustering algorithm. Section 5 discusses detailed experimental results. Finally, Section 6 gives concluding remarks and directions for future work.

2 Related Works

In this section, we discuss related works for clustering of sensor network. There are already a lot of works related to clustering network. LEACH [5] analyzes the performance of cluster-based routing mechanism with in-network data aggregation. LEACH leverages balancing the energy load among sensors using randomized rotation of cluster heads, but it does not guarantee good clustering head distribution. In ACE [13], the author clusters the sensor network in constant number of iterations using the node degree as the main parameter. In HEED [14], they show a distributed clustering algorithm which selects clusterheads based on residual energy of each node. However, they require too much iteration with transmitting the control packets. V. Mhatre and C. Rosenberg [11] formulated the required number of clusterheads and the required battery energy of nodes for both single hop and multi-hop communication mode.

GAF [6], SPAN [7], and ASCENT [8] tried to select a minimum number of nodes which join the multi-hop infrastructure. GAP, SPAN, and ASCENT share the same objective of using redundancy in sensor networks to turn radios on and off, thus prolong network lifetime. It does not guarantee optimal c clusterheads. In CLUSTERPOW [10], nodes are assumed to be non-homogeneously dispersed in the network. A node uses the minimum possible power level to forward data packets, in order to maintain connectivity while increasing the network capacity and saving energy.

Recently, T. J. Kwon *et al.* proposed passive clustering (PC) which does not run clustering algorithm periodically but uses piggybacking with on-demand routing in ad hoc network or sensor network [9]. In [12], authors proposed the division and merger based clustering in ad hoc networks. However, they did not consider the optimal number of clusters. The above two clustering algorithms intends to maintain the connectivity and to reduce the communication cost in networks in which nodes move. However, mobility of nodes in sensor network is rare. They do not take account for aggregating data and balancing energy consumption. Sensor networks require balancing the energy consumption, since data generated in a sensor network are too much for an end-user to process and data from close sensors may be highly correlated [4]. Thus methods for clustering the network well distributedly and combining data into a small set of meaningful information are highly required.

In this paper, we will show a clustering scheme which achieves the well distributed clustering over the whole network by using the nodes' remaining energy and the number of nodes within a cluster.

3 Analysis of Energy Consumption

In this section, to address our problem, we define simple network model and analyze the source of difference in energy consumption between nodes in case of cluster-based network. Then we propose a new clustering algorithm to balance energy consumption taking advantage of the analysis results.

Our network model is that sets of sensors are homogeneously dispersed on a regular field. Network has following properties:

(i) Network is consist of homogenous nodes.
(ii) The nodes are quasi-stationary after deployment.
(iii) All nodes have equal initial energy and similar capabilities.
(iv) Each node has a fixed number of transmission power levels and can control the level.
(v) Each node functions as either a clusterhead or an ordinary sensor node.

To analyze the main cause of difference in energy consumption between nodes, we build a simple network communication model similar to the one in [5] in which the amount of energy required to transmit a packet over distance d is given by $l + \mu d^k$, where l is the amount of energy spent in the transmitter electronics circuitry, while μd^k is the amount of energy spent in the RF amplifier to counter the propagation loss. Network consists of nodes, N, and the actual number of clusterheads, c, become clusterheads in each round. Each node become clusterhead once every $\frac{N}{c}$ round, receive data from its cluster member sensor, and transmit the aggregated data to the sink. Otherwise, nodes join the nearest clusterhead and transmit acquired sensor data to it every $\frac{N}{c} - 1$ rounds.

The nodes consume their energy differently in cluster-based network depending on their role: when they are clusterhead and when they are ordinary sensor nodes.

First, we analyze the maximum difference of energy consumption when nodes are clusterheads. If nodes are dispersed homogeneously and clustering is well distributed, one clusterhead contains $\frac{N}{c} - 1$ sensor nodes. The energy consumption of receiving data at clusterhead: $(\frac{N}{c} - 1)l$, the energy consumption of aggregating data: $(\frac{N}{c} - 1)DA$ where DA is the energy on data aggregation, the energy consumption of transmitting data to the sink: $l + \mu d^k$. The distance of the farthest node from the sink: d_{max}, the distance from the nearest node from the sink: d_{min}. The difference in energy consumption between two nodes in case of clusterhead, D_{CH}, is defined as follows:

$$D_{CH} \approx \mu({d_{max}}^k - {d_{min}}^k) \qquad (1)$$

We simply set the size of sensor field to [0, 0] to [A, A] and the sink located on the end of the sensor field. Since the maximum distance in this field is assumed $\sqrt{2}A$, the greatest difference in energy consumption is approximately $\mu(\sqrt{2}A)^k$.

Second, we calculate the greatest difference of energy consumption when nodes are ordinary sensor nodes. The energy consumption of each node as an ordinary node is $(\frac{N}{c}-1)(l+\mu d^k)$. The energy of consumption varies depending on the distance of the nearest $\frac{N}{c}-1$ nodes which become clusterheads in turn. Since we assume the size of sensor field is A^2 and nodes are dispersed homogeneously, the coverage of each node is $\frac{A^2}{N}$ and the radius of each node is $\frac{A}{\sqrt{\pi N}}$. The distance between two neighbor nodes is $2\frac{A}{\sqrt{\pi N}}$. The transmission range which contains $\frac{N}{c}-1$ nodes is as Fig. 1 (a).

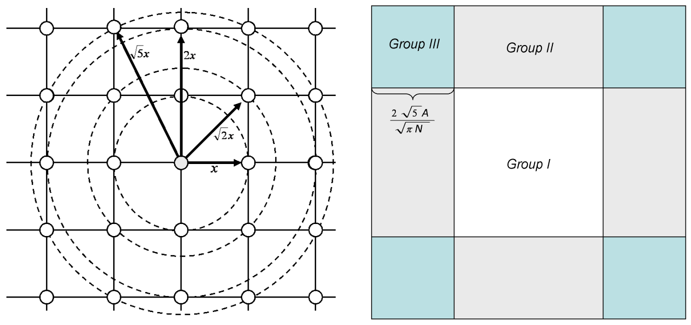

Fig. 1. (a)Numbers of neighbor per transmission range (b)Group classified by location within sensor field

We substitute x for $2\frac{A}{\sqrt{\pi N}}$. In case of transmission range x, $\sqrt{2}x$, $2x$, and $\sqrt{5}x$, the numbers of neighbor are 4, 8, 12, and 20 respectively. The numbers of neighbor node within certain range depends on the relative location within the sensor field. We categorize the nodes into three groups: the first group, called Group I, is located at the center of sensor field, the second group, called Group II, is located at which half of node's transmission range is out of sensor field, and the third group, called Group III, is located at which three-quarters of node's transmission range is out of sensor field. If the location of a node is $n_i = (x_i, y_i)$, N is 100, A is 100, and c is 5(5%), the transmission range is $\frac{2\sqrt{5}A}{\sqrt{\pi N}}$ and each group is as follows. We substitute α for $\frac{2\sqrt{5}A}{\sqrt{\pi N}}$. Fig. 1 (b) shows the location of each group.

- Group I: $(\alpha \leq x_i \leq (A-\alpha)) \wedge (\alpha \leq y_i \leq (A-\alpha))$
- Group II (two cases):
  $((0 \leq x_i < \alpha) \vee ((A-\alpha < x_i \leq \alpha)) \wedge (\alpha < y_i < (A-\alpha))$
  $((0 \leq y_i < \alpha) \vee ((A-\alpha < y_i \leq \alpha)) \wedge (\alpha < x_i < (A-\alpha))$
- Group III:
  $((0 \leq x_i < \alpha) \vee ((A-\alpha) < x_i \leq \alpha)) \wedge ((0 \leq y_i < \alpha) \vee ((A-\alpha) < y_i \leq \alpha))$

The most difference in energy consumption between Group I and Group III nodes, D_{ORD}, is defined as follows:

$$D_{ORD} \approx \mu \left(\left(\frac{\sqrt{40}A}{\sqrt{\pi N}} \right)^k - \left(\frac{\sqrt{10}A}{\sqrt{\pi N}} \right)^k \right) \qquad (2)$$

We simulate energy consumption of sensor network. Network consists of 100 nodes, network field is 100×100, every node start at same energy level, 2J, and the sink located from (50, 175) to (50, 225). Protocol runs clustering at every 20s and 5% of all nodes are selected as clusterheads using a heuristic algorithm, the simulated annealing [15], to build well-distributed clustering. After 20 rounds in which every node become a clusterhead at once, we get the residual energy of each node.

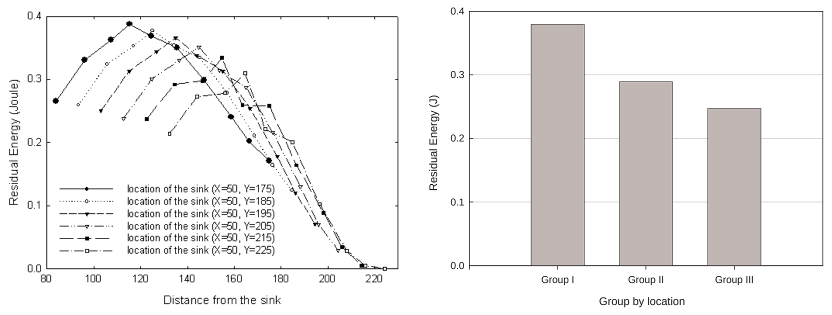

Fig. 2. (a)Residual energy per distance from the sink (b)Residual energy per group

Fig. 2 (a) shows the residual energy per the distance from the sink. The simulation results reflect that the distance between neighbors affects the energy consumption more than the distance between node and sink. The remaining energy does not decrease linearly since nodes in the middle of network have more neighbors than nodes in the fringe of sensor field. Fig. 2 (b) shows the average residual energy per above described group. Network consists of Group I: 24 nodes, Group II: 45 nodes, and Group III: 31 nodes.

In this section, we analyze the cause of difference in energy consumption between nodes in the case of single hop with clustering. We conclude that it is not effective that nodes become equally clusterhead in turn and node proximity to its neighbors affects difference more than distance between node and sink. Thus we choose the remaining energy of nodes, the distance between nodes within a cluster, and the number of nodes in a cluster as clusterhead selection criteria to balance energy consumption between nodes thus increase the amount of data delivery.

4 Proposed Clustering Algorithm

In this section, we describe our clustering algorithm, called Cluster-based self-Organizing Data Aggregation (CODA) in detail. First, we define the clustering problem we should address. Then, we propose a novel clustering algorithm based on previous analysis.

The clustering algorithm we introduce should meet the following requirements:

(i) Clustering is performed distributedly. Each node independently makes its decision based on local information.
(ii) Clustering terminates within a fixed number of iterations.
(iii) After clustering, each node is either a clusterhead or an ordinary node that belongs to one clusterhead.
(iv) Clusterheads are well-distributed over the network field.

Clustering is basically time-based operation. Thus all nodes in the network are synchronized and participate in the clustering at periodic intervals. In the pervious section, we show that clustering in which all nodes become clusterhead once in fixed period results in unbalanced energy consumption. Thus we do not restrict that all nodes should become clusterhead once in fixed period. We use the remaining energy of nodes, distance between nodes in cluster, and the number of nodes in a cluster as parameter for selecting clusterhead. The algorithm consists of three phases - the first elects clusterhead candidates based on the remaining energy and reelect clusterhead which have minimum distance cost within cluster ($Init$), the second merges clusters which have members below the lower threshold ($Merge$), and the third partitions clusters which have members over the upper threshold ($Partition$). In $Init$ phase, each sensor i elects itself to be a clusterhead at the beginning of $Init$ phase with probability $P_i(t)$. When there are N nodes in the network, $P_i(t)$ is as follows:

$$P_i(t) = \frac{E_i(t)}{\sum_{j=1(j\in N(i))}^{n} E_j(t)} \times \frac{n}{N} \times k \tag{3}$$

where $E_i(t)$ is node i's remaining energy, n is the number of neighbor in fixed transmission radius, and c is the expected number of clusters. If $P_i(t)$ is over the random number in [0, 1], node i is a candidate for clusterhead. Each candidate broadcasts advertisement message and then ordinary nodes join the nearest candidate. If ordinary node does not receive advertisement message, it become a clusterhead by itself. Candidate clusterhead calculates the intra-cluster cost for each member node. Intra-cluster cost for member node j is defined as follows.

$$Intra\_Cluster\_\_Cost_j = \sum_{k\in Cluster(i)} ((x_j - x_k)(x_j - x_k) + (y_j - y_k)(y_j - y_k))$$

Candidate clusterhead promotes the least cost node as new clusterhead if the

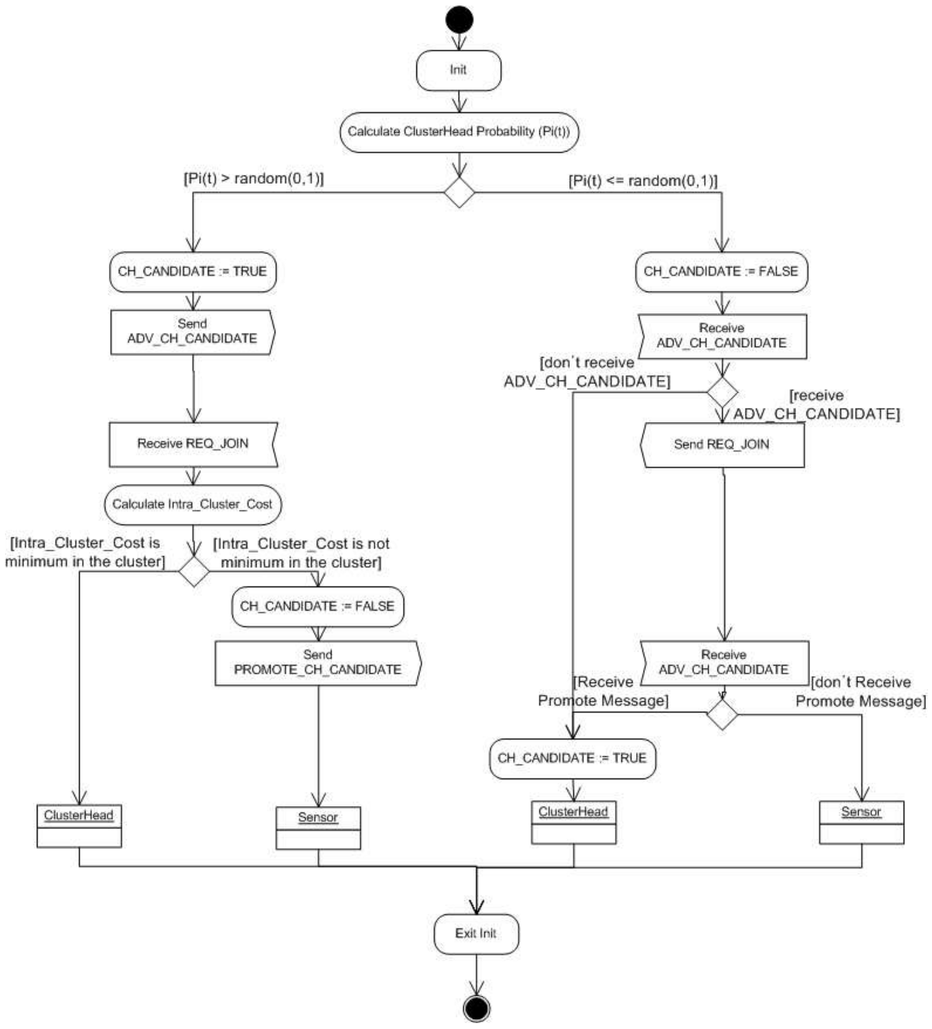
Init
Calculate ClusterHead Probability (Pi(t))
[Pi(t) > random(0,1)]
[Pi(t) <= random(0,1)]
CH_CANDIDATE := TRUE
Send ADV_CH_CANDIDATE
Receive REQ_JOIN
Calculate Intra_Cluster_Cost
[Intra_Cluster_Cost is minimum in the cluster]
[Intra_Cluster_Cost is not minimum in the cluster]
CH_CANDIDATE := FALSE
Send PROMOTE_CH_CANDIDATE
ClusterHead
Sensor
CH_CANDIDATE := FALSE
Receive ADV_CH_CANDIDATE
[don´t receive ADV_CH_CANDIDATE]
[receive ADV_CH_CANDIDATE]
Send REQ_JOIN
Receive ADV_CH_CANDIDATE
[Receive Promote Message]
[don´t Receive Promote Message]
CH_CANDIDATE := TRUE
ClusterHead
Sensor
Exit Init

Fig. 3. Flowchart of the *Init* procedure.

node's remaining energy is over the average in the cluster. The flowchart of this procedure is shown in Fig. 3.

We introduce the $Merge$ and $Partition$ phase since the probabilistic clusterhead election does not guarantee the expected number of cluster overall network and well-distributed cluster. After $Init$ phase, the $Merge$ phase starts if there is a cluster which has the member nodes below the lower bound ($Thresh_{Lower}$). A candidate clusterhead which has to be merged broadcast the request message (REQ_MERGE) and then neighbour candidate clusterheads which do not need to be merged reply (ACK_MERGE) to the request message. Then merged candidate clusterhead send the merge request message (REQ_CH_MERGE). If there is no response, it diverge two case. If the cluster has member nodes, it fixes it's $P_i(t)$ as 0.5 then processes the reelection. If the cluster do not have member nodes, it generates the random number in [0, 1] to decide whether it is still a clusterhead. When there is no neighboring candidate, it means that any nodes do not become the candidate clusterhead during previous $Init$ phase. The flowchart of $Merge$ is shown in Fig. 4 (a).

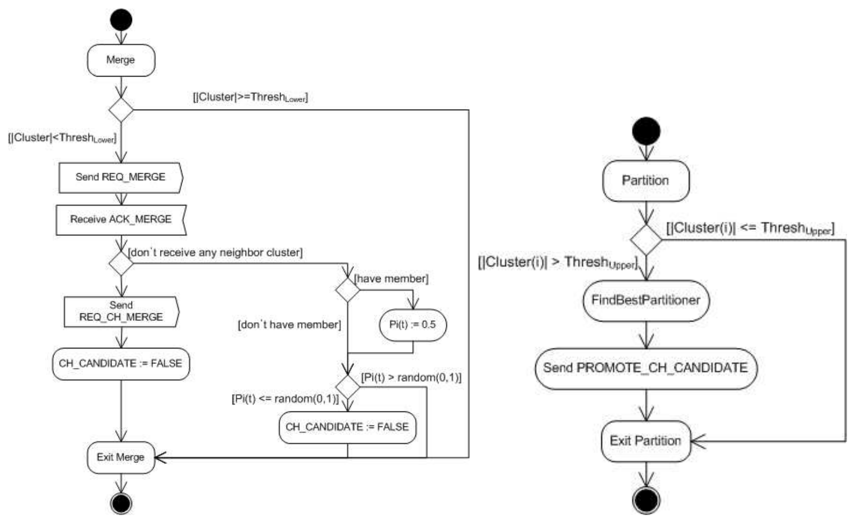

Fig. 4. (a)Flowchart of the $Merge$ procedure (b)Flowchart of the $Partition$ procedure.

After the $Merge$ phase, candidate clusterhead which has the number of member node over the upper bound ($Thresh_{Upper}$) partitions the cluster. A candidate clusterhead chooses an additional candidate clusterhead which partitions the cluster more evenly. The selection function, CH_{PART}, is defined as follow:

$$CH_{PART} = \arg \min_{j \in cluster(i)} ||Cluster(i)| - |Cluster(j)|| \qquad (4)$$

where $|Cluster(i)|$ is the number of nodes in cluster i. If the number of nodes in new cluster is below the lower bound ($Thresh_{Lower}$), existing candidate clusterhead do not progress the partition. This is to prevent clustering from going back to Merge condition. The worst case occurs when there is only one candidate clusterhead over the whole network. However, the Partition process will produce c clusters in $\log_2 k$. Therefore the clustering algorithm terminates in fixed iterations. The flowchart of $Partition$ is shown in Fig. 4 (a). To show the energy efficiency of the proposed algorithm, we simulate a system of 100 nodes on a 100 × 100 grid and analyze the results from several different points of view in the next section.

5 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm via simulation. We use the same parameters defined in [5], shown in Table 1.

Table 1. Simulation parameters

Parameter	Value
Network grid	(0,0) × (100,100)
Sink Position	(50, 175)
Threshold distance ($d_{crossover}$)	87m
ϵ_{elec}	50nJ/bit
$\epsilon_{friss-amp}(< d_{crossover})$	10pJ/bit/m^2
$\epsilon_{two-ray-amp}(\geq d_{crossover})$	0.0013pJ/bit/m^4
$\epsilon_{aggregation}$	5nJ/bit
Data packet size	500 bytes
Packet header size	25 bytes
Initial energy	2J
Number of nodes (N)	100
Number of clusters (c)	5
$Thresh_{Lower}$	$(N/c - (N/c)/2) = 10$
$Thresh_{Upper}$	$(N/c + (N/c)/2) = 30$

We measure two metrics to analyze the performance of clustering algorithm: **Network lifetime** measures the time of FND (First Node Die) and LND (Last Node Die). This metrics indicates how well balance the energy consumption over the whole network. **Data Delivery** is the total amount of data received at the sink. This metrics defines the network efficiency (Data received / Total energy spent). In addition, we observe the average number of cluster and average number of member.

We compare the CODA to the LEACH and Node-Weight (by remaining energy) clustering in terms of the above metrics. In LEACH every node becomes a clusterhead only once in $\frac{N}{c}$ round and Node-Weight selects a node which has the most remaining energy as a clusterhead in fixed radius. Fig. 5 (a) and Fig. 5

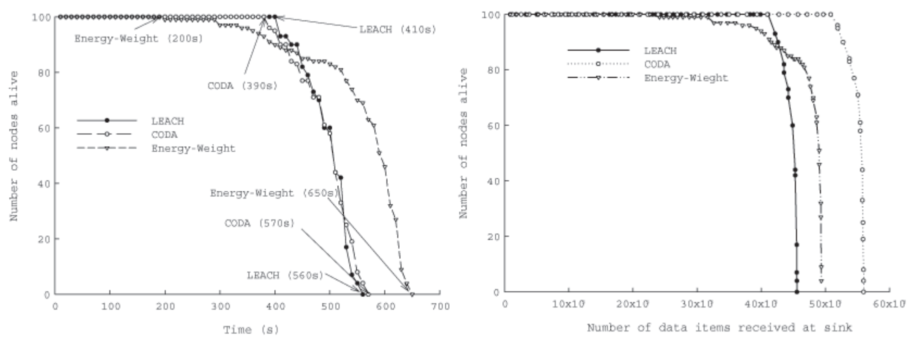

Fig. 5. (a)Number of nodes alive over time (b)Number of nodes alive per amount of data sent to the sink

(b) show the total number of nodes that remain alive over the simulation time and per amount of data received at the sink. While nodes remain alive for a long time in Node-Weight, time of FND (First Node Die) is earlier than any other method. This is because it keeps clusterheads having a regular distance but it consider only in fixed range. We see that nodes in CODA deliver 23% and 13% more data than LEACH and Node-Weight for the same number of node deaths respectively. Fig. 6 (a) and Fig. 6 (b) show the total number of data

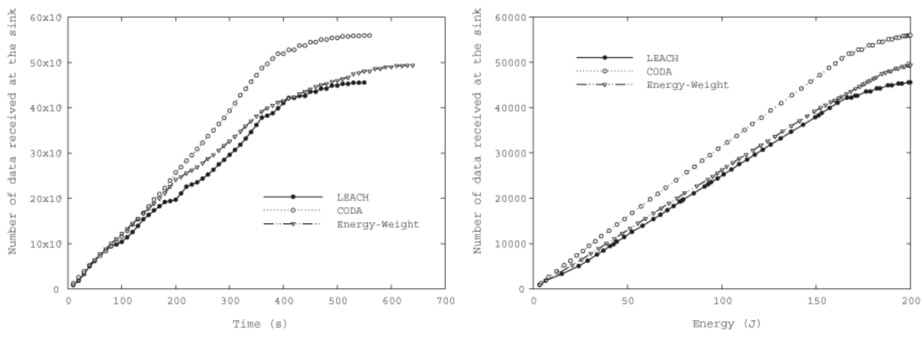

Fig. 6. (a)Total amount of data received at the sink over time (b)Total amount of data received at the sink per given amount of energy

received at the sink over time and for a given amount of energy. Fig. 7 shows that

CODA sends much more data to the sink in the simulation time than LEACH and Node-Weight. It indicates that CODA can build more efficient cluster over the whole network. Fig. 7 shows the reason why clustering performs differently.

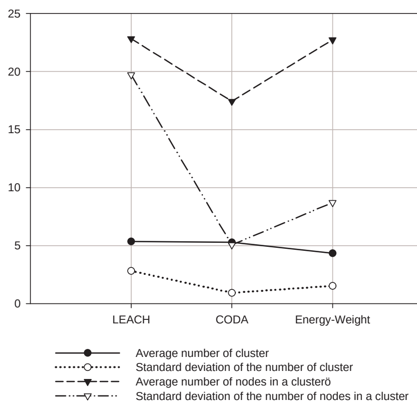

Fig. 7. Average number of cluster, Standard deviation of number of clusters, Average number of nodes in a cluster, and Standard deviation of number of nodes in a cluster.

Distance-based re-electing, merging, and partitioning enable CODA to achieve a better cluster than LEACH and Node-Weight.

6 Conclusions

In this paper, we have introduced a new energy-efficient clustering algorithm for wireless sensor networks. Our approach is to balance the number of nodes in a cluster. Clusterheads are randomly selected based on their residual energy and clusterhead migrates appropriately such that communication cost is minimized. Then cluster is merged and partitioned based on the number of nodes in a cluster. Simulation results show that CODA delivers more data than previous works. Although we have provided an efficient clustering algorithm, it performs only on single-hop network. We are currently incorporating CODA into a multi-hop routing model for sensor networks.

References

1. Akyildiz, IF., Weilian, S., Sankarasubramaniam, Y., Cayirci, E.: A Survey on Sensor Networks. IEEE Comm. Mag (2002) 102–114.
2. National Research Council: Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers. National Academy Press (2001)

3. Ganesan, D., Cerpa, A., Ye, W., Yu, Y., Zhao, J., Estrin, D.: Networking Issues in Wireless Sensor Networks. Journal of Parallel and Distributed Computing(JPDC) Special issue on Frontier in Distributed Sensor Networks (2003) 799–814
4. Estrin, E., Govindan, R., Heidemann, J., Kumar, S.: Next Century Challenges: Scalable Coordination in Sensor Networks. Proc. of the 5th Annual International Conference on Mobile computing and Networks (1999) 263–270
5. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: An Application-Specific Protocol Architecture for Wireless Microsensor Networks. IEEE Trans. on Wireless Comm. (2002) 660–669
6. Xu, Y., Heidemann, J., Estrin, D.: Geography-Informed Energy Conservation for Ad Hoc Routing. Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (2001) 70–84
7. Chen, B., Jamieson, K., Balakrishnan, H., Morris, R.: Span: an Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks. ACM Wireless Networks (2002) 85–96
8. Cerpa, A., Estrin, D.: ASCENT: Adaptive Self-Configuring Sensor Networks Topologies. Proc. of IEEE INFOCOM (2002) 1278–1287
9. Kwon, T., Gerla, M., Varma, V., Barton, M., Hsing, T.: Efficient Flooding with Passive Clustering-An Overhead-Free Selective Forward Mechanism for Ad Hoc Sensor Networks. Proc. of the IEEE (2002) 1210–1220
10. Kawadia, V., Kumar, P.: Power Control and Clustering in Ad Hoc Networks. Proc. of IEEE INFOCOM (2003) 459–469
11. Mhatre, V., Rosenberg, C.: Design guidelines for wireless sensor networks: communication, clustering and aggregation. Elsevier Ad Hoc Networks (2003) 45–63
12. Tomoyuki, O., Shinji, I., Yoshiaki, K., Kenji, I.: An Adaptive Multihop Clustering Scheme for Ad Hoc Networks with High Mobility. IEICE Trans. on Fundamentals (2003) 1689–1697
13. Chan, H., Perrig, A.: ACE: An Emergent Algorithm for Highly Uniform Cluster Formation. 2004 European Workshop on Sensor Networks (2004) 154–171
14. Younis, O., Fahmy, S.: Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach. IEEE INFOCOM 2004 (2004) 629–640
15. Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by Simulated Annealing. Science (1983) 13–22