Sensors **2007**, *7*, 730-742

sensors

ISSN 1424-8220

http://www.mdpi.org/sensors

Full Research Paper

Optimized Self Organized Sensor Networks

Sungyun Park [1], Kwangcheol Shin [1], Ajith Abraham [2] and SangYong Han [1,*]

[1] School of Computer Science and Engineering, Chung-Ang University 221, Heukseok-dong, Dongjak-gu, Seoul 156-756, Korea; E-mail: mouse500@ec.cse.cau.ac.kr kcshin@ec.cse.cau.ac.kr hansy@cau.ac.kr

[2] Center of Excellence for Quantifiable Quality of Service (Q2S), Norwegian University of Science and Technology, Norway; E-mail: ajith.abraham@ieee.org

* Author to whom correspondence should be addressed. hansy@cau.ac.kr

Received: 26 April 2007 / Accepted: 30 May 2007 / Published: 31 May 2007

Abstract: Wireless sensor networks are composed of a huge number of sensor nodes, which have limited resources - energy, memory and computation power. Energies are directly related to the lifetime of sensor network. If sensor nodes can be grouped to clusters, cluster member sensor nodes only need to communicate with cluster center (head) and this leads to energy conservation of the member sensors. So, how to compose clusters with minimal number of cluster heads, while including each node in a cluster is an important research issue. We propose a new advanced optimization algorithm for sensor network clustering. Using the proposed optimization algorithm, redundant cluster heads are eliminated, and unnecessarily overlapped clusters are merged. Optimization algorithm can be used as a clustering algorithm by itself and also manage the dynamic changes like node addition or die-out, while the network is even on the working state. We tested the proposed method as a clustering algorithm and compared it with two other recent sensor network clustering algorithms, Algorithm for Cluster Establishment (ACE) and Self Organizing Sensor network algorithm (SOS). The experiments results not only illustrate that the proposed algorithm could result in clusters with smaller number of cluster heads than others with any density of sensor networks, but also that the performance is more stable, which is also verified through repeated experiments.

Keywords: Optimization of clustering algorithm, Self Organizing Sensor algorithm, Intelligent Clustering, Wireless Sensor Network

---

1. Introduction

1.1 Related Research

From the mid 1990's, wireless sensor networks have been developed rapidly along with the growing development of micro devices of low-cost and wireless communication technology [1,2]. Sensor networks are usually composed of hundreds to myriads of sensor nodes, which appear to be sprinkled randomly by a car or airplane. Each node is equipped with a sensor to capture interesting information in certain area and a communication module to report it to the destination. They typically utilize intermittent wireless communication. Therefore, sensor networks should be well-formed to relay information to destination. Clustering is a fundamental mechanism to design scalable sensor network protocols. The purpose of clustering is to divide the network by some disjoint clusters. Through clustering, we can reduce routing table sizes, redundancy of exchanged messages, energy consumption and extend a network's lifetime [3]. By introducing the conventional clustering approach to the sensor networks, it provides a unique challenge due to the fact that cluster-heads, which are communication centers by default, tend to be heavily utilized and thus drained of their battery power rapidly.

Algorithm for Cluster Establishment (ACE) [4] clusters the sensor network within a constant number of iterations using the node degree as the main parameter. Self Organizing Sensor network (SOS) [3] illustrates that ACE performance relies on two parameters, which are usually manually adjusted according to the size and shape of a sensor network and they eliminate the use of manual parameters of ACE by selecting the primary head node and extend it to form other clusters. However, SOS has a structural weakness in sparsely distributed networks because it always needs linker node to form other clusters.

In the literature, besides ACE and SOS, there are some related works on forming and managing clusters for sensor networks. LEACH [5] rotates the role of a cluster head randomly and periodically over all the nodes to prevent early dying of cluster heads. Guru et al. [6] consider energy minimization of the network as a cost function to form clusters.

Krishnan and David Starobinski [7] used a message-efficient clustering, in which nodes allocate local "growth budgets" to neighbors. The algorithm produce clusters of bounded size and low diameter, using significantly fewer messages than the earlier, commonly used, expanding ring approach. They also presented a new randomized methodology for designing the timers of cluster initiators. This methodology provides a probabilistic guarantee that initiators will not interfere with each other.

Liu and Lin [8] introduce a re-clustering strategy and a redirection scheme for cluster-based wireless sensor networks in order to address the power-conserving issues in such networks, while maintaining the merits of a clustering approach. Based on a practical energy model, their simulation results show that the improved clustering method can obtain a longer lifetime when compared with the conventional clustering method.

When sensor nodes are organized in clusters, they could use either single hop or multi-hop mode of communication to send their data to their respective cluster heads. Mhatre and Rosenberg [9] presented a systematic cost-based analysis of both the modes, and provided guidelines to decide, which mode should be used for given settings. They also proposed a hybrid communication mode, which is a

combination of single hop and multi-hop modes, and which is more cost-effective than either of the two modes.

Younis et al. [10] present a novel approach for energy-aware management of sensor networks that maximizes the lifetime of the sensors while achieving acceptable performance for sensed data delivery. The approach is to set routes dynamically and arbitrate medium access in order to minimize energy consumption and maximize sensor life. The approach calls for network clustering and assigns a less-energy-constrained gateway node that acts as a cluster manager. Based on energy usage at every sensor node and changes in the mission and the environment, the gateway sets routes for sensor data, monitors latency throughout the cluster, and arbitrates medium access among sensors.

Pan et al. [11] considered a generic two-tiered wireless sensor network (WSN) consisting of sensor clusters deployed around strategic locations, and base-stations (BSs) whose locations are relatively flexible. Within a sensor cluster, there are many small sensor nodes (SNs) that capture, encode, and transmit relevant information from a designated area, and there is at least one application node (AN) that receives raw data from these SNs, creates a comprehensive local-view, and forwards the composite bit-stream toward a BS. Their research focus on the topology control process for ANs and BSs, which constitute the upper tier of two-tiered WSNs. By proposing algorithmic approaches to locate BSs optimally, they maximized the topological network lifetime of WSNs deterministically, even when the initial energy provisioning for ANs is no longer always proportional to their average bit-stream rate. By studying intrinsic properties of WSNs, authors established the upper and lower bounds of maximal topological lifetime, which enable a quick assessment of energy provisioning feasibility and topology control necessity.

In this paper, we propose a clustering optimization algorithm which can optimize the number of cluster heads produced by clustering algorithms like SOS or ACE. Suggested optimization algorithm also can work as a clustering algorithm itself and it can manage the dynamic changes like die out or addition of sensor nodes. Test results illustrate that the proposed algorithm can compose clusters with smallest number of heads regardless density of sensor network over SOS and ACE.

1.2 The Clustering Problem in Sensor Network

Clustering problem in sensor networks can be defined as follows: Let us assume that nodes are randomly dispersed in a field. At the end of clustering process, each node belongs to one cluster exactly and be able to communicate with the cluster head directly via a single hop [12]. Each cluster consists of a single cluster head and a bunch of member nodes as illustrated in Figure 1. The purpose of the clustering algorithm is to form the smallest number of clusters that makes all nodes of network to belong to one cluster. Minimizing the number of cluster head nodes would not only provide an efficient cover of the whole network but also minimizes the cluster overlaps. This reduces the amount of channel contention between clusters, and also improves the efficiency of algorithms that executes at the level of the cluster-head nodes [3,4].

Rest of the paper is organized as follows. In Section 2 the proposed optimization algorithm is illustrated. Experimental results are presented in Section 3 and some Conclusions are also provided towards the end.

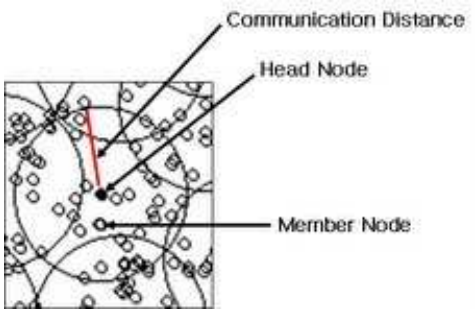


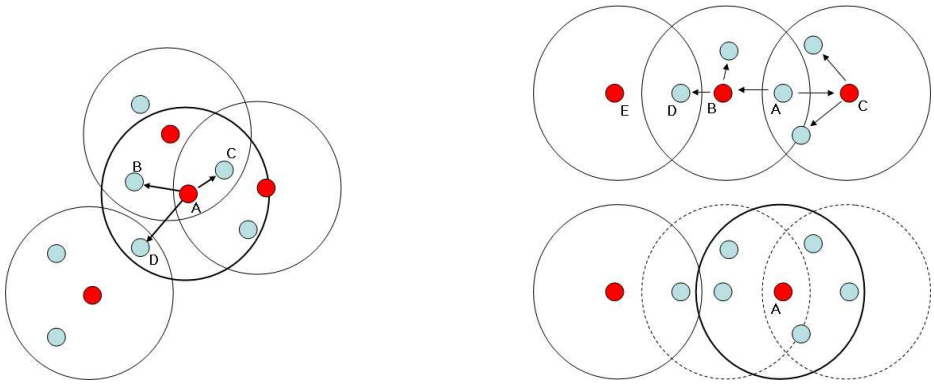
Figure 1. Clustering in a sensor network

2. Optimization Algorithm for Sensor Network Clustering

In this Section, we introduce the optimization processes for the clustered results of ACE and SOS, sensor network clustering algorithm, and we extend the optimization notions to dynamic changes of a sensor network like nodes die out or an addition and then we explain the proposed optimization algorithm, which can work as a clustering algorithm.

2.1 Dismiss and Merge Processes for Cluster Optimization

Once a sensor network is divided into disjoint clusters by the clustering algorithm, we apply the proposed optimization processes, which consists of dismiss and merge. 'Dismiss' is for removing redundant head node and 'Merge' is used in overlapped area of several clusters. Figure 2 shows simple examples of the situation which can be optimized. In the case of Figure 2 (a), the number of clusters can be reduced if node 'A' gives up its head position since all member nodes of node 'A' can be a member of other clusters including 'A' itself. Actually cluster of head node 'A' is redundant. In case of Figure 2 (b), we observe that two clusters can be merged into one. If node 'A' becomes a head node, all the members of head node 'B' and 'C', including 'B' and 'C' themselves also, can be members of new cluster 'A' or other existing clusters. These are the two basic cases we can reduce the number of cluster heads.



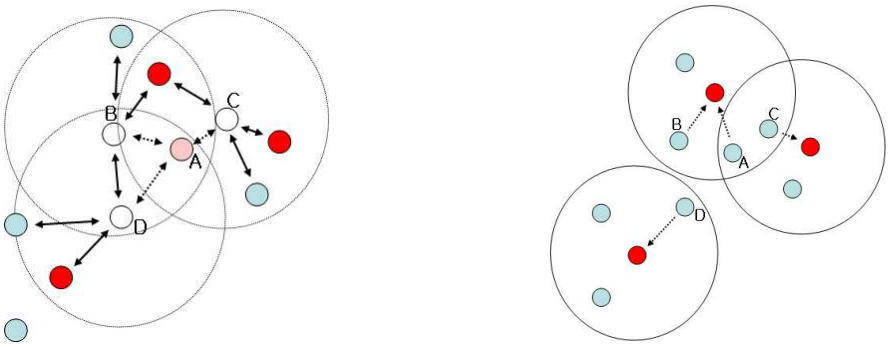
(a) (b)

Figure 2. Example of the cases can be optimized

2.2 Details in Dismiss and Merge Processes

Dismiss and merge are the basic operations of cluster optimization. This Section deals with the details of the processes in terms of interactions among nodes.

The process of eliminating redundant cluster heads, namely dismiss process, starts from a head node. A head node (for example, node 'A' in Figure 2(a)), which is triggered by its local random clock, broadcasts investigation message to its all members. The message is like this: "Is it all right if I give up being a head of you?" Members that receive the message from their head node examine their circumstances (Figure 3(a)). They figure out any head nodes in their communication range except the current head node. If they find at least one or more other head nodes, it means that they can be a member of other clusters instead of the current cluster head. Then the nodes send 'OK' message to the current head node. Otherwise send 'NOT OK'. The current head node decides based on the replies of its members. Only in the case that all replies are 'OK', the current head node confirms dismissal of its cluster. If the current head node sends a confirmation message to all member nodes, member nodes start to send join message to one of other clusters which they can join (Figure 3(b)).



(a) (b)

Figure 3. Example of 'Dismiss' process

The process of merging two clusters starts from a member node of a cluster. The member node 'A' triggered by its local random clock searches head nodes in its communication range (Figure 4(a)). If the member node 'A' recognizes that more than one head nodes exist except its head, it starts the process of investigation for merging. First, the member node chooses two head nodes including its current head node and sends a message like this: "Is it possible for me to be a head node instead of you two?" Two head nodes, which receive the message, also broadcast investigation message to their members with its ID and the ID of node 'A' which want to be a new head (Figure 4(b)). Member node that receives the investigation message replies back whether it can be a member node of the node which want to a new head or any head nodes in its communication range except current head node to be undesignated. This investigation process is the same as that of dismissal but with providing a ID of node which want to be a new head. Two head nodes gather replies from their member nodes and send it to the node 'A', which initially starts the process (Figure 4(c)). Node 'A' makes a decision. If all gathered replies from the two heads are 'OK' then the node 'A' changes its status to head and sends confirmation messages of merging to two head nodes, which will send the confirmation message of dismissal to its member nodes and change its status as a member of 'A'. Finally the member nodes which receive confirmation

message change their head to newly elected head ‘A’ or any head node in its communication range (Figure 4(d)).

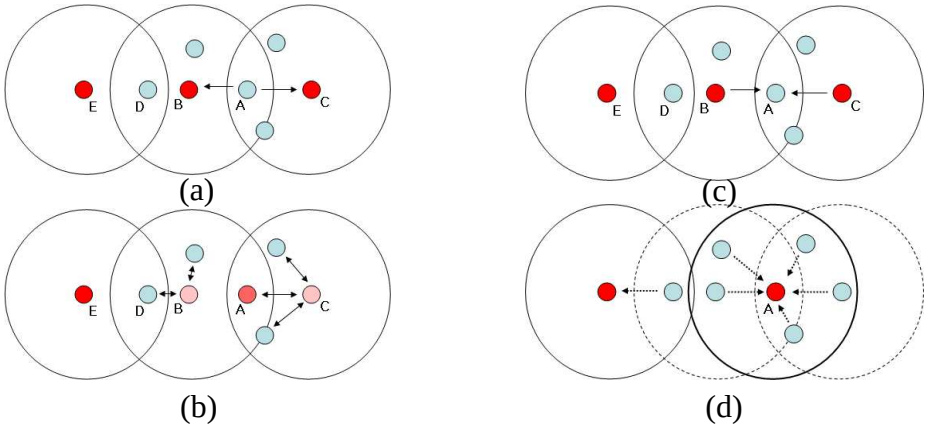


Figure 4. Example of ‘Merge’ process

2.3 Managing a Node Timer

These two processes start by local timer event. The timer’s value is set by random, so nodes in network wake up in random sequence just after network is activated. Timer event of a node makes one of two processes start according to the node’s state, head or member. Each node’s timer is activated at initialization time. Once a node’s timer is triggered, the timer is not reset until the node receives a message about membership change from other node. Membership change message occurs during the following 5 cases:

- a node’s state changes from a member node to a head node
- a node’s state changes from a head node to a member node
- a node changes its head node
- a node is being initialized.
- a node is going to die out

Through this random timer we expect that nodes wake up in random sequence, but one node can receive other investigation messages before the previous process ends. In this case the node which received two investigation messages should cancel the second one. Nodes, which receive cancel message, reactivate its timer in order to redoing the process later.

2.4 Using Optimization Algorithm as Clustering Algorithm

So far we have discussed about the proposed algorithm, which works for optimizing the sensor network already clustered. However, ours can also work as clustering algorithm by setting all nodes as head in network’s initial time and activates their timers as shown in Figure 5(a). If node A wakes up first, it operates ‘Dismiss’ process as following our optimization algorithm and finally it changes its status as member of node B (Figure 5(b)(c)). Like this way, the network can be clustered and optimized gradually by operating ‘Dismiss’ and ‘Merge’ processes without using other clustering algorithm.

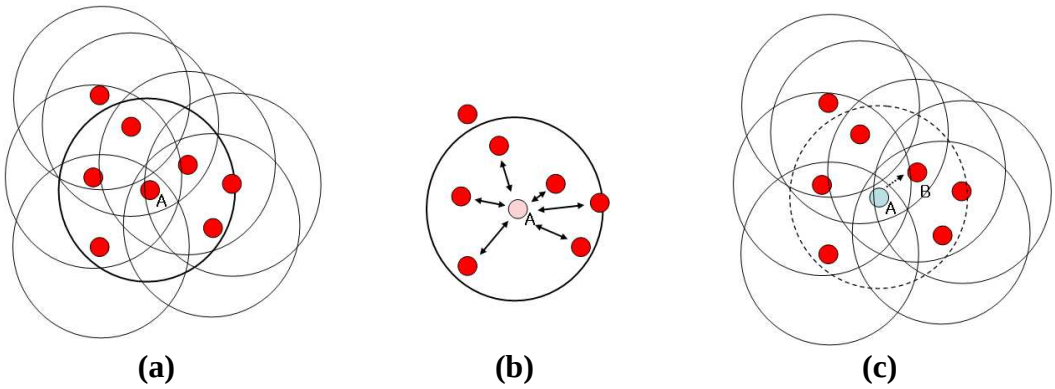


Figure 5. Example of using optimization algorithm as clustering algorithm

We define messages to communicate among nodes in Table 1 for the processes mentioned so far and Table 2 illustrates the pseudo code of our optimization algorithm executed in each node.

Table 1. Messages and their role among nodes.

Message	Description
SurveyForDismiss[oldHeads,newHeads]	Survey a chance of dismissal
Dismiss[oldHeads,newHeads]	Announce the confirmation of dismissal
SurveyForMerge[oldHEADs,newHEADs]	Survey a chance of merging
Merge[oldHEADs,newHEADs]	Announce the confirmation of merging
Join[nodeID]	Enroll to head node as member with nodeID
Activate	activate receiver's timer

* oldHeads : heads to be undesignated, newHeads : nodes to be designated as head

Table 2. Pseudo codes of optimization processes at each node

```
procedure Initialize
    myID := (unique ID)    myHead := myID    MEMBERS := {myID}
    broadcast Activate message to all nodes in communication range
procedure Finalize
    broadcast Activate message to all nodes in communication range
procedure MessageHandler
    TIMEOUT :       // this is sent by local timer when timed out
        if my status is head  node
            send  SurveyForDismiss[{myID},{}] messages  to MEMBERS
            wait for all replies
            if all replies are 'OK'
                send Dismiss[{myID},{}] message to MEMBERS
                MEMBERS := {}
        if my status is member node
```

```
        Head1 := myHead
        Head2 := select the nearest head node in my communication range except myHead
        if Head2 is not empty
          send SurveyForMerge [{Head1's ID, Head2's ID}, {myID}] to Head1 and Head2
          wait for all replies
          if all replies are 'OK'
            myHead := myID
            MEMBERS := {myID}
            send Merge [{Head1's ID, Head2's ID}, {myID}] to Head1 and Head2
            broadcast Activate to all node in communication range
SurveyForDismiss[oldHeads, newHeads] :
  newHeads := search head nodes  within my communication range in real heads or newHeads but not oldHeads
  if newHeads is empty
    reply NOT_OK
  else  reply OK
Dismiss[oldHeads, newHeads] :
  newHeads := search head nodes  within my communication range in real heads or newHeads but not oldHeads
  myHead := select the nearest HEAD node from newHeads
  send Join[myID] to myHead
  broadcast Activate message to all nodes in communication range
SurveyForMerge[oldHeads, newHeads] :
  send SurveyForDismiss[oldHeads, newHeads] to MEMBERS
  wait for all reply
  if all replies are 'OK'
    reply 'OK'
  else  reply 'NOT_OK'
Merge[oldHeads, newHeads] : send Dismiss[oldHeads, newHEADs] to MEMBERS but not to newHeads
Join[newMemberID] :  Add newMemberID to MEMBERS
Activate : activate timer with random time value
```

2.5 Supporting Abilities for Dynamic Changes in Sensor Networks

The proposed optimization algorithm can support dynamic changes, which can occur in real situation like nodes die out or during addition of new nodes.

Nodes may die and disappear from network due to some reasons such as energy depletion or system failure. In this case, the node broadcasts membership change message of its die out to the nodes in its communication range and correspondingly each node, which receives the message, activates its local timer and it will re-optimize the network soon.

Also nodes can be added to network. Because added nodes initialize themselves as a head node at their initializing time, these nodes declare themselves as head nodes and trigger membership change messages. All nodes which receive the message execute the optimization process, and the network will converge to its optimized state again. Figure 6 depicts that the newly added node can give effects to

change cluster head by merge process. These automatic optimizations in cases of deletion or addition of a node mean that our algorithm also works to maintain optimized state even during network's working time

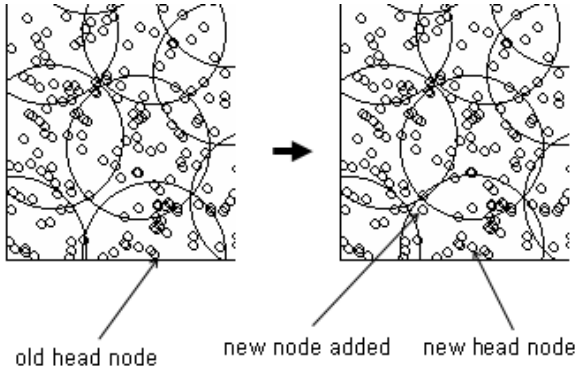


Figure 6. New added node can change cluster head.

2.6 Reducing Message Exchanges using Cache

The proposed optimization algorithm exchanges lots of messages due to the survey process. Exchanging message consumes energy of the sensor node. Energy problem is critical for a tiny sensor node. So we need to minimize the number of message changes even in clustering and optimization processes. We suggest a caching scheme to reduce the message exchanges. The optimization process we propose is divided into two phases, survey and execution. In survey phase, a node investigates other nodes in its communication range whether they can rejoin to other clusters and decides whether to execute dismiss or merge process based on the replies from them. In execution phase, based on the result of survey phase the optimization process is actually done. We can use the cache scheme in the survey phase when a change does not occur. If there is no membership change of all the nodes in a node's communication range, the node can cache the membership information of each node in its range. Fortunately, we know the membership changes of nodes in range from their notifications, because nodes report their membership change to all the nodes in communication range as we mentioned before. Nodes cache the membership information until the next notification of the nodes. With cache, nodes can save energy for survey phase. Figure 7 shows the energy consumption details during clustering process of our algorithm. It represents the average number of transmitted messages at one node during survey and execution phase. As experimental result shows, lots of energy in survey phase can be saved by the cache strategy.

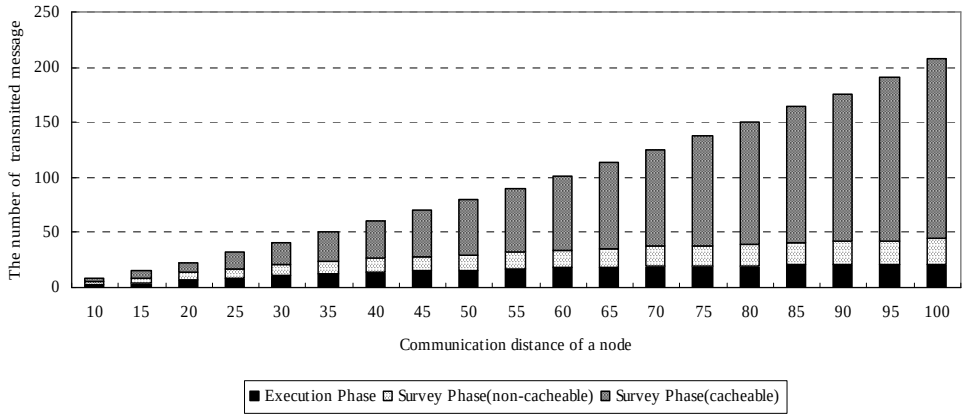


Figure 7. The average number of message exchanges of a node for clustering a network.

3. Experimental Results

3.1 Results of Clustering Algorithms

For the experiments, we randomly spread out 2500 sensor nodes in 500x500 rectangle area and the communication range of each node is adjusted from 4 to 100. We also implemented the SOS and ACE algorithms to compare with the proposed approach. All the three algorithms were run 250 times by randomly changing the position of each node and the average value is computed. Figure 8 shows the detailed clustering results. It is noticed that SOS algorithm depicts poor performance for small communication ranges (4~16). In other words, SOS cannot work efficiently when nodes are distributed sparsely (Figure 9 also illustrates this). For communication ranges within 18~80, SOS has improvements compared to ACE. For denser distribution of nodes (82~100), SOS shows slightly better results than SOS. The proposed algorithm has better overall performance when compared to SOS and ACE.

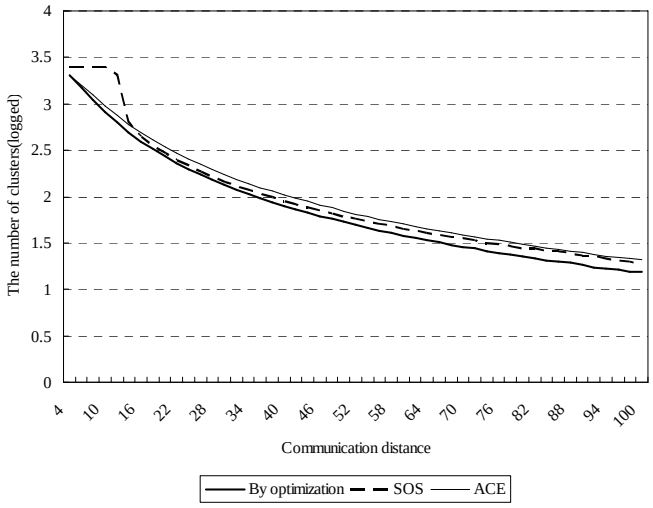
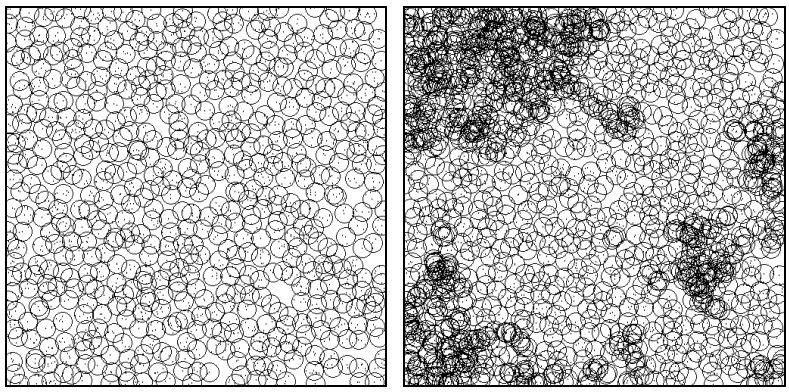


Figure 8. Performance results of three clustering algorithms.



(a) By optimization (613 cluster heads) (b) SOS (1386 cluster heads)

Figure 9. Clustering results of the proposed algorithm and SOS approach. (Distance is 12)

3.2 Stability of Clustering Algorithms

Figure 10 shows the variance of the number of clusters for 250 experiment trials. The small variance value of an algorithm means that it gives uniform results regardless of the network's density. SOS shows a high variance with small communication ranges, which means SOS is not stabilized with sparse node distribution. In most of areas, the proposed algorithm shows more stabilized results than others.

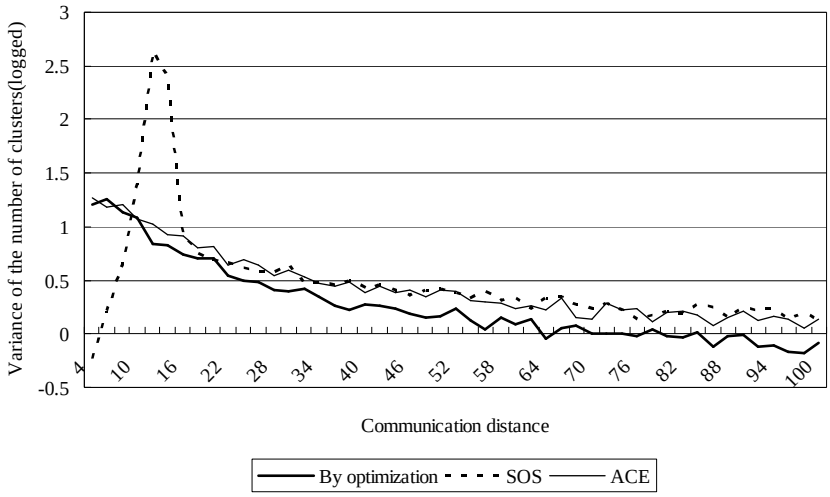


Figure 10. Variance of the number of clusters

3.3 Execution Time of Clustering Algorithms

Figure 11 shows the average execution time of each algorithm for 250 experiment trials with 2500 nodes in 500x500 rectangle area. ACE shows best performance by completing it in constant time regardless the size of networks. Our optimization method used as clustering algorithm takes the worst execution time but in the case of applying ours as optimization method to the result of ACE, ours shows much faster. Though our optimization method takes the worst time complexity in our simulating

environment, but in real situation where each sensors work concurrently, we expect ours show better execution time than this.

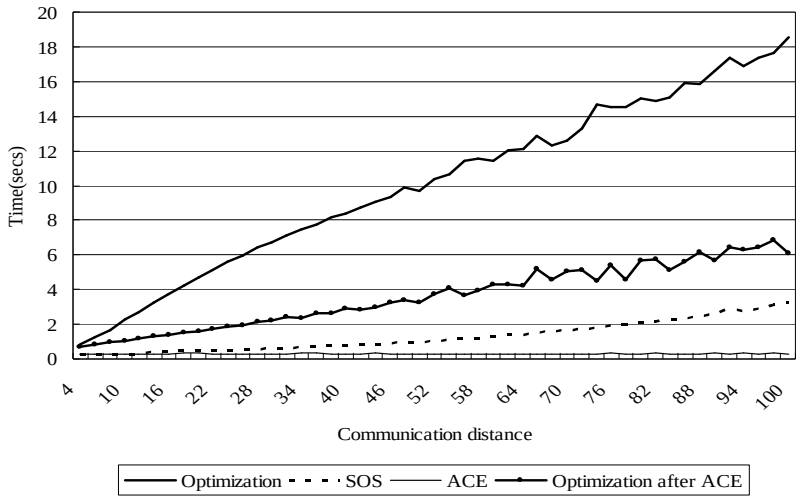


Figure 11. Comparison of execution times

4. Conclusions

In this paper, we presented a new clustering algorithm for minimizing the number of cluster heads. It is impossible to grasp the optimal number of clusters of a sensor network without exhausted search, which is impractical. We introduce the methods which can merge unnecessarily overlapped clusters and dismiss redundant cluster heads to get close the optimal number of clusters. Through experiments, we illustrated the improvements of the proposed optimization algorithm. We compared it with two recent algorithms, ACE and SOS. We also illustrated the clustering results and the stability of each algorithm. From the results, it is evident that the proposed optimization algorithm works better and is more stabilized than other two algorithms. We also suggested simple caching strategy to reduce the number of message exchanges in survey process.

As future research, we plan to extent our optimization algorithm to take into account of the residential energy of each node also, while selecting the cluster head.

Acknowledgements

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the Chung-Ang University HNRC-ITRC (Home Network Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

References and Notes

1. Akyildiz, I. F.; Su, W.; Sankarsubramaniam, Y.; Cayirci, E. Wireless Sensor Networks : a survey. *Computer Networks* **2002**, *38*, 393-422.
2. Kahn, J. M.; Katz, R. H.; Pister, K. S. Next Century Challenges : Mobile Networking for "Smart Dust". *Proceedings of Mobicom* **1999**, 271-278.

3. Shin, K.; Abraham, A.; Han, S. Y. Self Organizing Sensor Networks Using Intelligent Clustering. *Lecture Notes in Computer Science* **2006**, *3983*, 40-49.
4. Chan, H.; Perrig, A. ACE: An Emergent Algorithm for Highly Uniform Cluster Formation. *In 2004 European Workshop on Sensor Networks* **2004**, 154-171.
5. Heinzelman, W.; Chandrakasan, A.; Balakrishnan, H. An Application Specific Protocol Architecture for Wireless Microsensor Networks. *IEEE Transactions on Wireless Communications* **2002**, *1* (4), 660-670.
6. Guru, S. M.; Hsu, A.; Halgamuge, S.; Fernando, S. An Extended Growing Self-Organizing Map for Selection of Clusters in Sensor Networks. *International Journal of Distributed Sensor Networks* **2005**, *1* (2), 227-243.
7. Krishnan, R.; Starobinski, D. Efficient clustering algorithms for self-organizing wireless sensor networks. *Ad Hoc Networks* **2006**, *4* (1), 36-59.
8. Liu, J. S.; Lin, C. H. R. Energy-efficiency clustering protocol in wireless sensor networks. *Ad Hoc Networks* **2005**, *3* (3), 371-388.
9. Mhatre, V.; Rosenberg, C. Design guidelines for wireless sensor networks: communication, clustering and aggregation. *Ad Hoc Networks*, **2004**, *2* (1), 45-63.
10. Younis, M.; Youssef, M.; Arisha, K. Energy-aware management for cluster-based sensor networks. *Computer Networks* **2003**, *43* (5), 649-668.
11. Pan, J.; Cai, L.; Hou, Y. T.; Shi, Y.; Shen, S. X. Optimal base-station locations in two-tiered wireless sensor networks. *IEEE Transactions on Mobile Computing*, **2005**, *4* (5), 458 – 473.
12. Younis, O.; Fahmy, S. Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach. *In Proceedings of IEEE INFOCOM* **2004**, 629-640.