DACA: Data-Aware Clustering and Aggregation in Query-Driven Wireless Sensor Networks

Somaieh Bahrami, Hamed Yousefi, and Ali Movaghar
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
{Somayeh_bahrami,hyousefi}@ce.sharif.edu, movaghar@sharif.edu

***Abstract*—Data aggregation is an effective technique which is introduced to conserve energy by reducing packet transmissions in wireless sensor networks (WSNs). In addition, it is possible to consume less energy by using the spatial correlation and redundancy of data in dense networks to form clusters of nodes sensing similar values and, in turn, transmit one data packet per cluster. In this paper, we propose a Data-Aware Clustering and Aggregation scheme (DACA) to manage the energy constraint in a query-driven WSN. The DACA selects cluster head nodes by forming a new factor as a function of three parameters including the residual energy, the data value, and the number of neighbors at each node. Moreover, it exploits a cluster merging method to overcome the problem of the previous studies in which all sensor nodes become cluster heads over time, so it prolongs the network lifetime. Extensive simulations in NS-2 verify the superiority of our approach.**

***Index Terms*—Wireless Sensor Networks; Data Aggregation; Data-aware Clustering.**

I. Introduction

The development of tiny, low-cost, and low-power sensor nodes which are able to read data from the environment, process it, and communicate in short distances, has led to the emergence of WSNs [3]. A WSN contains a large number of sensor nodes densely deployed in an environment to collect information about the physical phenomena by using continuous queries on a tree-based routing structure rooted at the sink node.

As the main source of power in a sensor node is a limited and irreplaceable battery, it is necessary to have an energy-efficient design for WSNs. Since communication and data transmission consume more energy than data processing, a portion of data processing can be pushed inside the sensor nodes equipped with the unit processors. By using this method, which is called in-network processing, the amount of data that should be transmitted inside a WSN will be reduced.

Data aggregation is a promised technique leveraging in-network processing to reduce energy consumption in the monitoring applications. In this technique, instead of sending the received packets separately, each node first combines all the incoming packets with its own read data into a packet, and computes an intermediate result according to some aggregation functions such as average, sum, minimum, and maximum, then transmits the partial result as a fixed size packet to the next node. By using data aggregation, the number of transmitted packets is reduced and as a result, the cost of communication decreases.

In addition, because the sensor nodes are deployed densely in the environment, it is possible that the adjacent nodes sense the similar or redundant data at the same time. By leveraging the existing spatial data correlation and redundancy among sensor nodes, it is possible to form clusters of nodes sensing similar values from the environment within a given threshold.

Extensive research has been conducted to cluster the entire network. However, there is a limited study on data-aware clustering methods. CAG [9] utilizes the correlation to form clusters in a tree-based structure sensor network. Only the cluster head nodes send their own data as the representative of the other cluster members. However, it has a major drawback; the adjacent nodes are not aware of each other's status, so it is highly probable that redundant clusters are formed. To cope with this, DEDAC [10] proposes a data- and energy-aware backoff algorithm, but the problem of building unnecessary clusters still remains. Moreover, in both CAG and DEDAC, during the cluster adjustment which is done to solve the inconsistency of sensor readings over time, a cluster head node cannot become a cluster member and this role remains unchanged. This issue leads to the formation of clusters with one member, so after a while all sensor nodes will become cluster heads.

In this paper, we propose a Data-Aware Clustering and Aggregation scheme (DACA) to efficiently conserve energy of a query-driven WSN. Our work has a twofold contribution. First, the DACA selects cluster head nodes by forming a new factor as a function of three parameters including the residual energy, the data value, and the number of neighbors at each node. This results in a sequence of cause and effect: a significant reduction in the number of clusters, reducing the number of transmissions, and efficient energy conservation in WSNs. Second, the DACA exploits a cluster merging method to prevent forming of one-member clusters, so it prolongs the network lifetime. The results reveal that the proposed algorithm outperforms other existing schemes in terms of network lifetime and energy consumption.

The rest of the paper is organized as follows. Section II discusses the related work. Section III explains the proposed scheme and presents its specifications. Simulation results are presented and discussed in Section IV. Finally, Section V concludes our work.

978-1-4673-1544-9/12/$31.00 ©2012 IEEE

II. Related Work

The TAG algorithm [8] uses the data aggregation technique in a tree-base structure network to reduce the number of transmitted data packets. In recent years, several data aggregation approaches which use the existing data correlation among sensor nodes have been proposed to decrease the traffic of data collection and conserve energy. In [6], centralized and distributed EGCD algorithms which use data correlation to select a subset of connected sensor nodes, in a way that the data of these nodes is used to reconstruct the data of other sensor nodes are proposed. After that, only the selected nodes are involved in data collection. The centralized approach is more accurate than the distributed scheme, but it is not a scalable technique. In [4], a Grid-based Spatial Correlation Clustering algorithm (GSCC) forms the sensor nodes with high similarity of data values in the same clusters. After the cluster formation, the cluster head node which is located nearer to the center of the grid approximates the data of the cluster members according to historical data, and then only the cluster head sends the aggregated data to the sink. This algorithm is based on data with Gaussian distribution and it suffers from message overhead during the cluster formation. An aggregation architecture which is composed of a static routing tree and a dynamic clustering structure in [5] is proposed. In this work the clustering method is LEACH-C [7] without taking into consideration of the residual energy of nodes. In addition, the standard complete-link agglomerative grouping approach is used to construct correlation groups in each cluster. During data collection in each group, a few nodes as representative nodes transmit their data to the associated cluster head which uses the linear regression to predicate the data of the other nodes. The proposed algorithm is based on data with linear distribution, and the performance of this approach depends on the number of representative nodes in each group.

For a query driven sensor network, the Clustered AGgregation (CAG) approach is proposed in [9] to use data correlation for clustering. This algorithm provides approximate results with bounded error for user queries. CAG operates in query, response, and cluster adjustment phases. While the query routing tree is built in the first phase according to the TAG algorithm [8], the clusters are formed by using a specified error threshold in the user query. During the response phase only the cluster head node transmits its own sensed data, and the partial aggregation results are computed in the intermediate nodes along the tree. The cluster adjustment phase is done in cluster member nodes periodically. If the difference between the data of a cluster member and the associated cluster head is bigger than a specified error, the cluster member node must either join a neighboring cluster or form a new one. In CAG, it is highly possible that redundant clusters are formed in the query phase because the neighboring nodes are not aware of the status of each other.

DEDAC [10] is another clustering approach which uses the spatial correlation among sensor nodes' data to form clusters. DEDAC uses a data- and energy-aware backoff algorithm to solve the problem of redundant cluster formation in CAG. However, since clusters are not distributed in the network by the backoff algorithm, it is still possible that unnecessary clusters take form. In DEDAC, to reduce the cluster adjustment overhead, the size of a cluster is limited by k-hop, but this issue can lead to an increase in the number of clusters. Moreover, during the response phase, data aggregation operations are only performed in the cluster head nodes, and the other intermediate nodes just forward data packets. This, in itself, results in more energy. During the cluster adjustment phase, the cluster member nodes do not have the chance to migrate to a neighboring cluster, so like the cluster head election process these nodes use the backoff algorithm. Finally, like CAG, DEDAC suffers from forming one-member clusters.

III. DACA Protocol

In this section, we introduce the network model, DACA as a new data-aware clustering protocol, and details of different phases used for efficient data collection in query-based WSNs.

A. Network Model

A network system consists of static and battery-powered sensor nodes as well as one sink node. Sensor nodes are distributed randomly in a reliable environment and have the same physical capabilities. Besides, each sensor node learns the number of its one-hop neighbors via Hello packets. The sink node is responsible for receiving the user query, propagating it through a TAG-like tree structure, and then collecting the results from the sensors. The user query is an aggregation query used for continuous monitoring of the physical phenomena like temperature, light, etc. in definite time epochs. Moreover, this query includes a specified error threshold which indicates the expected accuracy of the final result.

B. Algorithm Overview

DACA is a clustering approach which uses the data correlation among sensors to form clusters in a query-driven network, while only the cluster head nodes contribute to the result collection. It uses the data value and the number of neighbors to form fewer and larger clusters, eliminate more sensor nodes during the result collection, and consequently save much more energy. DACA consists of five phases including: 1) query dissemination and cluster formation, 2) cluster head election, 3) result collection, 4) cluster adjustment and maintenance, and 5) cluster merging which will be explained in the following parts.

1) Query Dissemination and Cluster Formation: The query packet has a format like this:

$Q =< A, Agg, \tau, ParentID, MyID, level, CHData, CHNi >$

In this format, A is the attribute in which the user is interested for data collection and Agg is the aggregation function. The τ field indicates the user-provided error threshold. In addition, the $ParentID$, $MyID$, and $level$ fields are used to build the query routing tree. In this tree, the sink node is

not only the root, but also the first node which is elected as the cluster head. As mentioned above, DACA exploits data correlation to form clusters. To achieve this goal, DACA uses the data difference between the cluster head nodes and the others. Therefore, it is necessary for a sensor node to learn the data of the current cluster head node in a way that the *CHData* field is used to store it. Once a sensor node receives the query packet, it compares its own data value with the value of the *CHData* field. If condition (1) is met, it will join the current cluster and will broadcast the query packet with its own value of *ParentID*, *MyID* and *level*; otherwise, the sensor node will enter the cluster head election phase.

$$SensorData \in \{CHData \pm CHData \times \tau\} \tag{1}$$

Moreover, the *CHNi* field contains the number of neighbor nodes of the current cluster head, and is used during the cluster head election phase.

2) Cluster Head Election: In this phase, the sensor nodes which did not join the current cluster compete to form new clusters with their own data. This competition is based on a backoff algorithm. When such a sensor node receives the query, it does not broadcast it immediately. A timer at the node is initialized by factor t_n as a function of three parameters which include the residual energy, the data value, and the number of neighbors as follows:

$$t_n = \alpha \times e_n + (1 - \alpha) \times (d_n \times ni_n) \tag{2}$$

where α is the normalization factor, and we investigate its impact by carrying out simulation experiments in Section IV. Since the cluster head nodes consume much more energy than the others, the node with more residual energy is more suitable to become a cluster head. In this equation, e_n is used as the residual energy factor, and calculated by:

$$e_n = 1 - \frac{E_{n-current}}{E_{n-initial}} \tag{3}$$

where $E_{n-current}$ and $E_{n-initial}$ are the residual energy and initial energy of the current node, respectively. Moreover, as we mentioned before, the main goal of DACA is the reduction of the number of clusters to decrease the number of transmitted data packets. The greater the data value of the node, the larger the interval length $[NodeData \pm NodeData \times \tau]$. Finally, the larger the intervals as well as the larger the number of neighbor nodes can lead to the dominance of more numbers of adjacent nodes becoming the members of the mentioned interval. This, in itself, results in larger clusters. The impact of the data value is considered in d_n which is calculated according to:

$$d_n = \frac{d_{current-cluster}}{d_{n-current}} \tag{4}$$

where $d_{current-cluster}$ and $d_{n-current}$ indicate the data value of the current cluster head node and the sensor node, respectively. Finally in (2), ni_n is used to investigate the effect of the number of neighboring nodes, and is computed as follows:

$$ni_n = \frac{ni_{current-cluster}}{ni_{n-current}} \tag{5}$$

where $ni_{current-cluster}$ and $ni_{n-current}$ represent the number of neighboring nodes of the current cluster head node and of the current sensor node, respectively.

As it is evident, the sensor node with more residual energy, larger data value, and more neighboring nodes has priority to become a cluster head, and win the competition. If a sensor node which is waiting for the expiration of its timer receives a new query packet meeting condition (1), it will leave the competition and become a member of the new cluster, then it broadcasts the query; otherwise, it forms a new cluster and broadcasts the query packet including its own data value and the number of neighbor nodes. Thus, using this backoff mechanism can also solve the problem of redundant cluster formation in the neighboring nodes.

3) Result Collection: After propagating the user query, the result is collected periodically. As mentioned before, only the sensor values from the cluster heads take part in computing the final result. During this phase, partial aggregated results are computed in both the cluster head and bridge nodes to reduce the number of the transmitted data packets. Bridge nodes are used to communicate between different clusters. They just aggregate the incoming data packets, and then forward the partial aggregated results to their parent nodes in the query tree. During the transmission of the results, each node updates its neighbor table by snooping on the broadcast medium.

Moreover, to guarantee the expected accuracy in the final result, it is necessary to calculate the size of the cluster. In CAG and DEDAC algorithms, during the query dissemination and cluster formation, each cluster member node transmits a control packet separately containing its ID and the ID of its cluster head. Each cluster head estimates the size of its own cluster by counting the number of the received control packets. To reduce the overhead of the cluster size calculation, DACA uses a different approach during the first collecting result epoch after the cluster formation. In this approach, control packets are piggybacked on data packets and a counting aggregation function is applied on the value of the control packets in the intermediate nodes. It should be noted that this has to be done after the cluster adjustment or merging phase.

4) Cluster Adjustment and Maintenance: The purpose of the cluster adjustment and maintenance phase is to match the current data dynamics and make cluster membership consistent as sensor readings change over time. Each cluster member node periodically does this phase in every adjustment interval depending on the data correlation.

During the cluster adjustment, if the difference between the data value of the cluster member and the associated cluster head is bigger than a given error threshold, the cluster member node must change its membership. In this situation, if the data difference between the sensor node and one of the neighbor clusters is smaller than the threshold, the sensor node joins it; otherwise, it waits to receive a cluster adjustment packet from other nodes by using a backoff algorithm. If the timer

expires and the sensor node cannot join a new cluster, it will form a new cluster with its own data value. In both cases, the node must inform its children which are in the same cluster. Therefore, it will broadcast the cluster adjustment packet with a format like the query packet. Once the children receive this cluster adjustment packet, they enter the cluster adjustment and maintenance phase and repeat the above-mentioned process. During this phase, the structure of the routing tree changes. Specifically, when a sensor node receives a cluster adjustment message from a node which is different from its parent, it updates its parent node and its level in the tree with new values.

5) Cluster Merging: As mentioned before, CAG and DEDAC have a major drawback: during the cluster adjustment phase, a cluster head cannot become a cluster member and this role remains unchanged, so all sensor nodes become cluster heads over time. Therefore, after a while, the entire network takes part in the result collection, so the limited energy resource is wasted and the network lifetime reduces. To cope with this problem, DACA exploits a cluster merging method. This phase is performed periodically and simultaneously with the cluster adjustment phase in cluster head nodes.

During this phase, each cluster head uses its neighbor table to become informed of the data value of its neighbor cluster heads in upper levels on the routing tree. If the data difference between its value and one of the upper neighbor cluster heads is smaller than the specified error threshold, this cluster head will change its role, update its level in the routing tree, and join the neighbor cluster as a cluster member. After that, this node must announce its new role and the data value of the new associated cluster head to its member nodes. Once the members receive the cluster merging message, they enter the cluster adjustment phase to select a suitable cluster according to the data correlation information. It should be noted that the cluster adjustment and merging begin from the top levels of the tree. Moreover, the adjustment (merging) of one cluster has no effect on the adjustment (merging) of the other clusters.

IV. Performance Evaluation

In this section, we will evaluate the performance of our proposed algorithm, DACA, via the NS-2 simulator [2]. The goal of the simulation is to indicate that DACA can outperform two other data-aware clustering algorithms, CAG [9] and DEDAC [10], by comparing the network lifetime, the energy consumption, the number of transmitted data packets, and the number of formed clusters over time. In our simulation, we used a real data set of Intel lab [1] containing 54 Mica2Dot sensor nodes. The simulation parameters are shown in Table I.

Figure 1 shows the number of formed clusters in the DACA algorithm for various error thresholds (τ = 0.03, 0.06, 0.09) under different values of α parameter. As we can see, the best result is obtained when the value of α is 0.3. Therefore, in all simulation scenarios, we used the value of 0.3 for this parameter.

To compare the number of formed clusters during query dissemination in DACA, CAG, and DEDAC, we change the

TABLE I
Simulation Parameters

TX Power	54.4 mW
RX Power	28.8 mW
Sleep Power	< 3 μW
TX Range	8 m
Initial Energy	5 J
Sample Period	Every 31 s
Query Duration	1 day $\approx$ 2880 epochs
Adjustment and Merging Intervals	Every 310 s
Attribute	Temperature
Aggregation Function	AVG

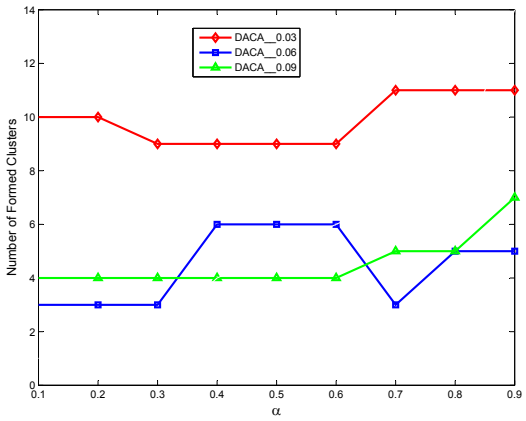


Fig. 1. Number of formed clusters under different values of α parameter

error threshold in the interval [0.01, 0.1]. Since in DEDAC the size of a cluster is limited by k hops, we repeated its simulation for different values of k (3, 5, 7, and ∞). As it can be seen in Figure 2, when we use DACA, the number of formed clusters is on average 45% and 37% smaller than that of CAG and DEDAC, and this indicates the performance of our proposed timer which takes three important parameters into account for cluster formation using a backoff mechanism. We can see that by reducing the value of k in DEDAC, the number of formed clusters will increase. In the following simulations, we choose the value of 7 for k in DEDAC. Moreover, we repeated the simulation scenarios for different values of error threshold but we only include the results for the value of 0.03 due to the space limitation in the paper.

Figure 3, shows the formed clusters by GAG, DEDAC (7), and DACA algorithms in the Intel lab. In this figure, the cluster head nodes are shown by the black circles and the root of the query routing tree (Node 42) is located in the bottom left corner. As we can see DACA forms lager clusters, and this, in itself, results in the reduction of the number of cluster heads. For example, when nodes 40 and 41 receive the query message, they cannot join the cluster in which the cluster head is node 42 since the data difference among them is not bounded by the threshold value of 0.03. However, according to the data set values, there is a potential for nodes 40 and 41 to form one cluster jointly. As we can see in CAG, these two nodes form separate clusters immediately since they are not aware of the status of each other. By using the backoff algorithm in

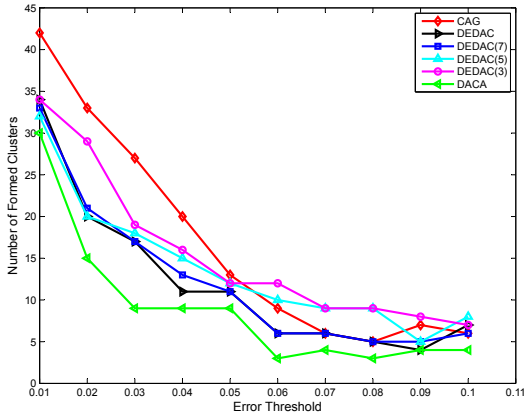
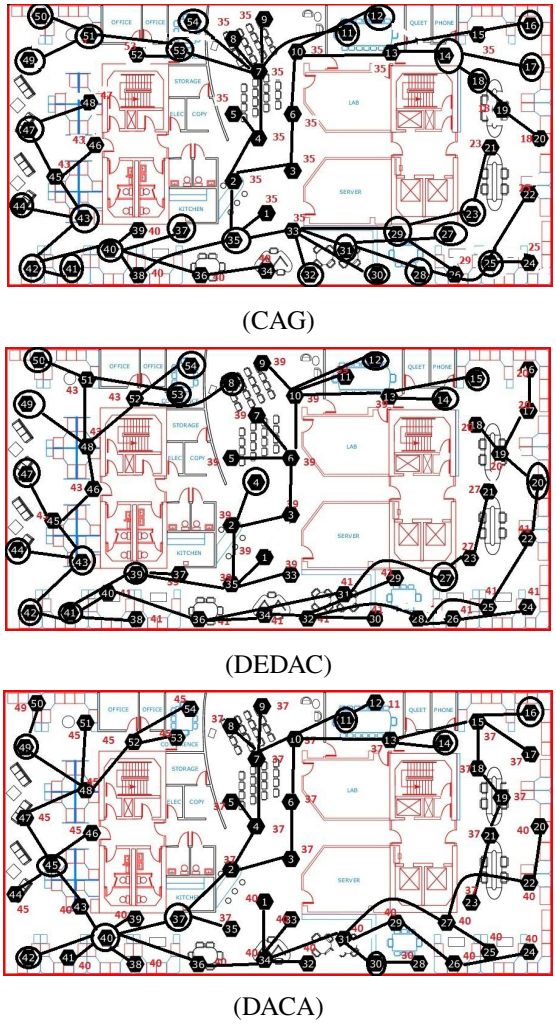


Fig. 2. Number of formed clusters during query dissemination under different values of error threshold

DEDAC and DACA algorithms these nodes join each other in one cluster; In DEDAC node 41 becomes a cluster head and forms a cluster with fourteen members while by using DACA, node 40 is chosen as a cluster head and forms an eighteen-member cluster.

In addition, we can see that the adjacent nodes which have correlated data do not form separate clusters in DACA. For example, the data difference among nodes 49 and 50 is bounded by the threshold value of 0.03; therefore, these nodes can form one cluster jointly. As we can see in DACA, these nodes form one cluster, and node 49 is selected as the cluster head. Although DEDAC uses the backoff algorithm, these adjacent nodes cannot be timely aware of the status of each other, so their timers expire before they can receive the query message of each other and, in turn, form separate clusters.

To compare the network lifetime, we computed the number of alive nodes in every epoch of result collection. In Figure 4, DACA outperforms CAG and DEDAC in terms of network lifetime. As it is evident, after 298 epochs there exist some dying nodes when we use DEDAC, and in 500 epochs the energy of the entire network is depleted. On the other side, when by using CAG the number of active nodes is fewer than 3 in 2670 epochs, by using DACA all the sensor nodes are alive. Thus, DACA prolongs the network lifetime more than 2300 and 440 epochs compared to DEDAC (7) and CAG, respectively; since in DEDAC (7) unlike CAG intermediate nodes forward the incoming data packets separately without applying an aggregation function, these nodes consume much more energy. This, in itself, results in a reduction of the network lifetime. Besides, DACA forms fewer clusters than CAG while merging the adjacent clusters according to spatial correlations over time, so it reduces the number of transmitted data packets, and consequently, energy consumption; that's why nodes battery power lasts longer. Since by using DEDAC (7) and CAG the network can collect the result of the query until 298 and 2222 epochs, respectively, we divide the later simulation into two parts, in which we compared DACA with DEDAC (7) and CAG from epoch 1 to 298, while only with



(CAG)

(DEDAC)

(DACA)

Fig. 3. Intel laboratory - Clusters formed by CAG, DEDAC, and DACA algorithms

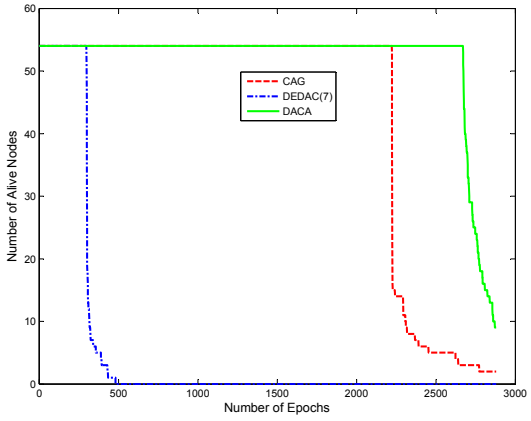
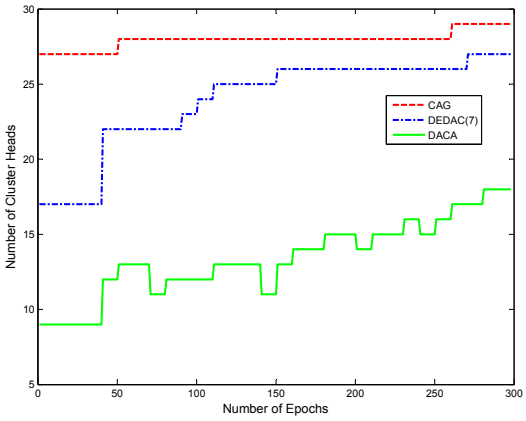
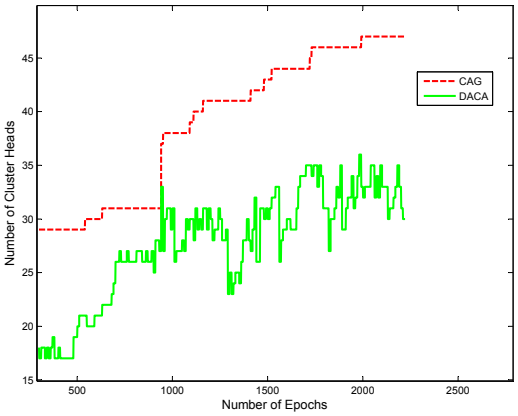


Fig. 4. Number of alive nodes per epoch

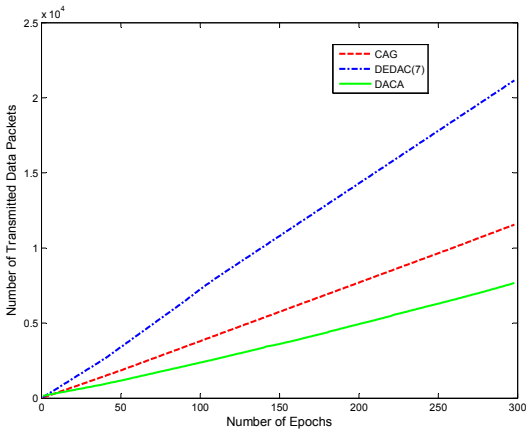


(a)

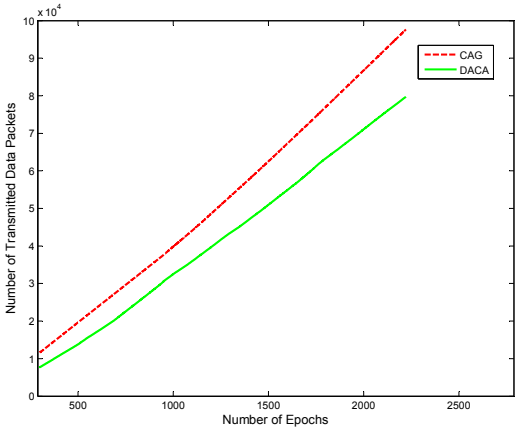


(b)

Fig. 5. Number of cluster heads per epoch



(a)



(b)

Fig. 6. Number of transmitted data packets per epoch

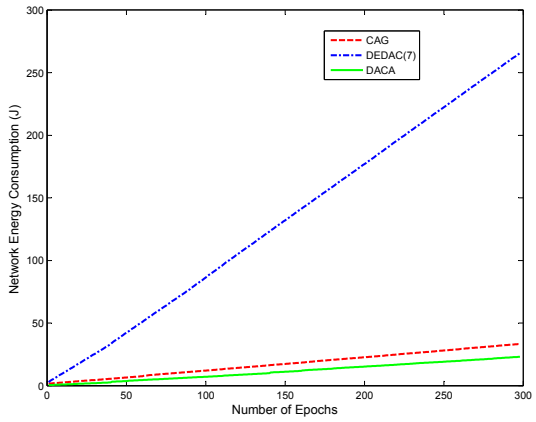
CAG from epoch 299 to 2222.

Figure 5 shows the number of cluster heads in each epoch. We can see that the number of cluster heads always increases in CAG and DEDAC while it fluctuates in DACA because of using a cluster merging method. According to Figure 5a, while there are 18 cluster heads in the network by DACA, CAG and DEDAC (7) form 29 and 27 cluster heads, respectively. Specifically, until epoch 298, DACA has an average reduction of 52% and 44% in terms of the number of cluster heads compared to CAG and DEDAC (7), respectively. As we can see in epoch 2222 (Figure 5.b), the total numbers of cluster heads are 30 and 47 in the network by DACA and CAG, respectively. Thus, DACA outperforms CAG on average by 29% from epoch 299 to 2222 while it generally has 32% fewer cluster heads than CAG.

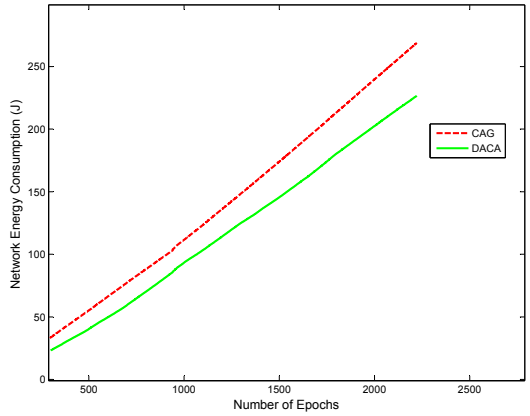
Due to major contribution of packet transmissions in overall network energy consumption, we use the number of transmitted data packets to show the energy efficiency of the proposed solution. As we can see in Figure 6, DACA outperforms CAG and DEDAC (7) on average by 23% and 64%, respectively. Actually, DACA not only forms fewer numbers of clusters but also prevents all nodes from becoming cluster heads over time by using an efficient merging method, so it significantly decreases the number of transmitted data packets while using data aggregation in all intermediate nodes. As we mentioned before, during the result collection phase, only cluster heads transmit their own data. Therefore, regarding Figure 5 we expect that this number by DEDAC to be smaller than that of CAG, but something else is observed. The reason is that in DEDAC bridge nodes do not use data aggregation, so the large number of data packets is transmitted in the network directly.

Furthermore, in Figure 7, we can see the actual energy costs associated with the algorithms per epoch. Until the epoch 298, DACA has an average reduction of 40% and 91% compared to CAG and DEDAC (7), respectively. Furthermore, DACA totally outperforms CAG by 21% in terms of energy consumption. As previously stated, packet transmissions are the major contributors in overall network energy consumption and hence Figure 7 exhibits the same trend as Figure 6.

Here, we want to show that the reduction of the number of cluster heads, or in fact the reduction of the number of nodes whose own data participates in computing the final result, does not have a negative effect on data accuracy. To measure data accuracy, we compute the relative error in the achieved results as below:

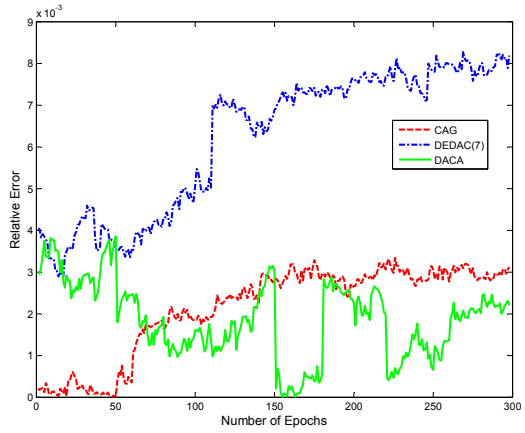


(a)

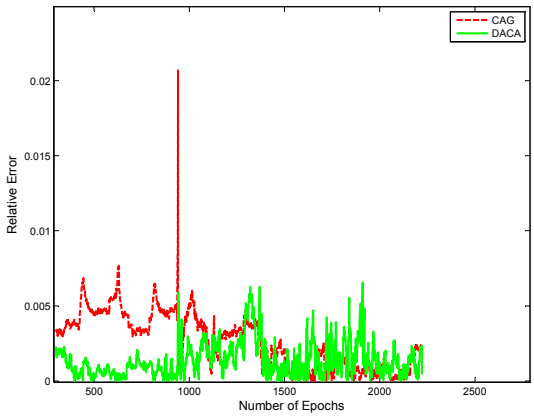


(b)

Fig. 7. Network energy consumption per epoch



(a)



(b)

Fig. 8. Relative error per epoch

$$RelativeError = \frac{|EstimatedResult - ExactResult|}{ExactResult} \quad (6)$$

where *EstimatedResult* is the achieved results by DACA, CAG and DEDAC, and *ExactResult* is the accurate one which can be obtained by TAG where all network nodes participate in data collection. As we can see in Figure 8, although the number of cluster heads is reduced much more by DACA compared to CAG and DEDAC (7), the relative error is always less than the specified error threshold.

V. Conclusion

In this paper, we proposed an adaptive clustering algorithm, DACA, based on data correlation among sensor nodes while using data aggregation during result collection in query-driven WSNs. It exploits a backoff algorithm as well as a cluster merging method to form fewer and larger clusters and save much more energy. Results evaluated in the simulation on a real data set demonstrated a significant performance improvement in terms of network lifetime, energy consumption, number of cluster heads, and number of transmitted data packets.

References

[1] "Intel Lab Data , http://db.csail.mit.edu/labdata/labdata.html."
[2] "The network simulator, http://www.isi.edu/nsnam/ns."
[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug. 2002.
[4] Z. Chen, S. Yang, L. Li, and Z. Xie, "A clustering approximation mechanism based on data spatial correlation in wireless sensor networks," in *Proceedings of WTS'10*, April 2010, pp. 1–7.
[5] C. Cho, C. Lin, Y. Hsiao, J. Wang, and K. Yang, "Data aggregation with spatially correlated grouping technique on cluster-based WSNs," in *Proceedings of SENSORCOMM'10*, July 2010, pp. 584–589.
[6] H. Gupta, V. Navda, S. Das, and V. Chowdhary, "Efficient gathering of correlated data in sensor networks," *ACM Transactions on Sensor Networks*, vol. 4, no. 1, pp. 1–31, Feb. 2008.
[7] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, Oct. 2002.
[8] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: A tiny aggregation service for ad-hoc sensor networks," in *Proceedings of OSDI'02*, 2002.
[9] S. Yoon and C. Shahabi, "The Clustered AGgregation (CAG) technique leveraging spatial and temporal correlations in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 3, no. 1, March 2007.
[10] Y. Zhang, H. Wang, and L. Tian, "Energy and data aware clustering for data aggregation in wireless sensor networks," in *Proceedings of MASS'07*, Oct. 2007, pp. 1–6.