Ad Hoc Networks 8 (2010) 328–344

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Using hierarchical agglomerative clustering in wireless sensor networks: An energy-efficient and flexible approach

Chung-Horng Lung *, Chenjuan Zhou

Department of Systems and Computer Engineering, Carleton University, Ottawa, Ontario, Canada K1S 5B6

ARTICLE INFO

Article history:
Received 10 December 2008
Received in revised form 7 July 2009
Accepted 29 September 2009
Available online 2 October 2009

Keywords:
Wireless sensor networks
Clustering-based routing
Hierarchical agglomerative clustering
Energy efficiency

ABSTRACT

In wireless sensor networks (WSNs), hierarchical network structures have the advantage of providing scalable and resource efficient solutions. To find an efficient way to generate clusters, this paper adapts the well-understood hierarchical agglomerative clustering (HAC) algorithm by proposing a distributed HAC (DHAC) algorithm. With simple six-step clustering, DHAC provides a bottom-up clustering approach by grouping similar nodes together before the cluster head (CH) is selected. DHAC can accommodate both quantitative and qualitative information types in clustering, while offering flexible combinations using four commonly used HAC algorithm methods, SLINK, CLINK, UPGMA, and WPGMA. With automatic CH rotation and re-scheduling, DHAC avoids reclustering and achieves uniform energy dissipation through the whole network. Simulation results in the NS-2 platform demonstrate the longer network lifetime of the DHAC than the better-known clustering protocols, LEACH and LEACH-C.

1. Introduction

Wireless sensor networks (WSNs) become an invaluable research area by providing a connection between the world of nature and that of computation by digitizing certain useful information. The WSNs are rather different from the conventional wireless networks due to the following constraints: (i) a large number of sensor nodes often needs to be randomly deployed to reduce cost, (ii) sensor nodes are often deployed in unreachable harsh environments. The sensor nodes may fail, causing communication failures and consequent network topology changes, (iii) data rates in WSNs are much lower than the conventional wireless networks, (iv) most applications of sensor networks have highly asymmetric communication links; the collected data is sent up to a particular node which is called a sink, and (v) sensor nodes are often required to have ultra low power consumption.

One of the most important challenges of WSNs design is develop a method or protocol so that the randomly deployed numerous sensor nodes behave in a collaborative and organized way. Each sensor node wants to maximize its own utility function. In addition, the entire network needs balance in resource assignments to perform in a way that is useful and efficient. Network routing protocol design becomes far more critical to WSNs performance than that of from conventional communication networks.

Among numerous proposed network routing protocols in past years, hierarchical routing protocols greatly contribute to system scalability, lifetime, and energy efficiency [2]. To our best knowledge, all current clustering protocols are top-down approaches, which first formulate a global knowledge of a WSN, specifying but not detailing the first-level nodes. Based on the global knowledge of network and predefined methods, the protocols first build the upper level of clusters by selecting certain nodes as CHs. Then they group the rest of the nodes into the designated cluster as cluster members. Many algorithms randomly select CHs, which usually results in low cluster quality.

* Corresponding author. Tel.: +1 613 520 2600x2642.
E-mail addresses: chlung@sce.carleton.ca (C.-H. Lung), cjzhou@sce.carleton.ca (C. Zhou).

1570-8705/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2009.09.004

The motivation of our research is to provide efficient clustering without requiring the global knowledge of network by reversing the clustering approach from top-down to bottom-up. With the bottom-up approach, sensor nodes collaborate and build clusters before they select CHs. In this manner, the bottom-up approach can be a better way to implement self-organization, scalability and flexibility. By carefully inspecting the relationships among nodes, we develop a simple and flexible bottom-up clustering scheme, DHAC, to suit different application scenarios.

This paper is an extension of our previous work [17,30]. The main idea is to tailor the HAC [3,21] algorithm for WSNs to efficiently generate hierarchical clusters. Our research was inspired by the fact that the HAC has been successfully applied to many disciplines, and it is a conceptually and mathematically simple approach. Additionally, the HAC algorithm offers flexible options depending on different available data. Specifically, this paper improves the HAC to a distributed algorithm, DHAC.

Unlike previous hierarchical protocols, DHAC provides innovational schemes for the clustering and cluster maintenance. First, DHAC is a bottom-up approach that forms clusters by only requiring one-hop neighbor information. The limitation of the sink is eliminated. Second, DHAC can be executed with or without knowledge of location (quantitative data and qualitative data). Section 5 presents DHAC with different input data types, quantitative location data, quantitative Received Signal Strength (RSS) data, and qualitative connectivity data. Third, DHAC provides higher energy efficiency. DHAC performs clustering only once, at the initial stage. Hence, DHAC can avoid the time and energy consumed by reclustering. In the cluster maintain stage, DHAC uses automatic CH rotation and re-scheduling to ensure uniform energy dissipation within clusters. DHAC further avoids unnecessary energy consumption of re-scheduling by considering the network traffic.

The rest of the paper is organized as follows. Section 2 highlights of related work. Section 3 briefly illustrates HAC methods. Section 4 presents DHAC that is tailored for WSNs. Section 5 demonstrates experiments and simulation results. Finally, section 6 is the conclusions.

2. Background and related work

2.1. Network routing protocols

Network routing protocols are responsible for the network structure and routing scheme. Many researchers have proposed routing solutions for WSNs. The proposed routing protocols can be broken down into different groups based on assorted criteria [2,1,18]. Network structure, resource awareness, and protocol operation method are basic taxonomies of WSN routing protocols. For example, RRCH [19], AHP [25] and HEED [26] are the hierarchical protocols based on network structure. HEED is also an energy-aware protocol when considering resource awareness. In this paper, we focus on flat and hierarchical routing schemes based on network structure.

In a flat network, all nodes are typically assigned an equal role and functionality. The desired data are sent out to the network through multi-hop routes. To eliminate many redundant transmissions through the network, flat protocols focus on how to route based on the application queries. Most flat protocols are data-centric and ensure nodes only transmit the valuable data which match the query attributes. In many cases, flat protocols result in more complicated routing because of the large scale and dynamic network topology of WSNs. Sensor protocols for information via negotiation (SPIN) [12] and directed diffusion (DD) [16] protocols are important flat protocols which motivated the design of many other protocols that follow similar concepts.

In hierarchical networks, nodes are separated to play different roles, such as CHs and cluster members. The higher level nodes, cluster head (CHs), manage the grouped lower level nodes (cluster members) and collect data from them. Each CH collects data from the cluster members within its cluster, aggregates the data, and then transmits the aggregated data to the sink. All of the hierarchical routing protocols aim at selecting the best CH and clustering the nodes into appropriate clusters in order to save energy. Since the CHs have responsibility for the collecting, aggregating, and transmitting data over longer distances to the sink, they consume more energy compared to the other cluster members. The hierarchical clustering protocol may execute reclustering and reselecting of CHs periodically in order to distribute the load uniformly among the whole network.

Although hierarchical protocols have their native weaknesses such as requiring time synchronization, potential producing non-optimal routing, and utilizing higher overhead for cluster management, they reveal attractive advantages by dealing with the constraints in WSNs. Compared with flat protocols, hierarchical protocols offer a more feasible solution to handle large-scale networks with their enhancements to better share limited wireless channel bandwidth, balancing node energy consumption and reduce communication expense [2,1,27].

2.2. Related work on hierarchical protocols

The literature is rich with various methods and protocols to support WSNs. The reader is referred to [2,1,28, 27,18] for details. The section concentrates on the hierarchical protocols. By the method of CH selection, the hierarchical routing protocols can be classified into two categories: random-selected-CH protocol and well-selected-CH protocol. The former randomly selects CHs and then rotates the CH task among all nodes, while the latter carefully selects appropriate CHs and then gathers nodes under the CHs based on the network status.

2.2.1. Random-selected-CH protocols

2.2.1.1. LEACH. LEACH [13,15] protocol is proposed to balance the energy dissipation in sensor networks. The main idea of LEACH is that sensor nodes can be randomly selected as CH based on their previous experiences of being a CH. In the cluster formation phase, each sensor node generates a random number between 0 and 1. Each sensor node has its threshold which is related to the predefined percentage of CHs in a network. If the generated random number is less than the threshold, then the node becomes the CH; otherwise, it joins a cluster to be a cluster member. How to calculate the threshold is the key of LEACH.

After clusters are set up, the CH broadcasts a transmission schedule within the cluster and asks its members to send data based on a TDMA approach. In the steady phase, CHs are responsible to aggregate and send data to the sink. After a certain period of time spent in the steady phase, the network goes to formation phase to redo the clustering. LEACH uses the periodic reclustering to alleviate the deterioration of cluster quality.

LEACH is completely distributed and requires no global knowledge of the network. LEACH clustering terminates within a constant number of iterations but it does not guarantee good CH distribution and assumes uniform energy consumption for CHs. Furthermore, the idea of dynamic clustering brings extra overhead, e.g., head changes and advertisements, which may diminish the gain in energy consumption.

2.2.1.2. BP. Anker, et al. [4] adopts the belief propagation (BP) algorithm based on the probabilistic graph model to iteratively compute marginal probabilities on trees by local message passing. The method considers performance of a multi-hop network. Performance was evaluated against HEED [26] using the TinyOS simulator [22]. The paper shows that the reclustering process is less frequently triggered using the approach with the expense of high initial clustering overhead. Overall, the clustering scheme based on the BP method is more efficient.

2.2.1.3. ERA. Energy Residue Aware (ERA) [10] clustering algorithm is another energy-aware hierarchical approach. It is also improved from LEACH by including the communication cost into the clustering. The communication cost includes residual energy, communication energy from the CH to the sink and communication energy from the cluster members to the CH. There is a difference from HEED: ERA uses the same CH selection scheme as LEACH but provides an improved scheme to help non-CH nodes choose a "better" CH to join by calculating the clustering cost and finding CH according to maximum energy residue.

2.2.1.4. RRCH. RRCH [19] performs cluster formation only once to avoid the high energy consumption during clustering phase. RRCH uses a similar method to LEACH to set up clusters. Once the clusters are set up, RRCH keeps the fixed clusters and uses the round-robin method to choose the node to be the CH within the clusters. Every node has a chance to be CH during a frame. When a node has been detected as an abnormal node, the CH modifies the scheduling information and broadcasts it to the entire cluster during frame modification; then its cluster members delete the abnormal node based on the received schedule information.

RRCH has the same defect of LEACH: no guarantee of cluster quality. Without the periodic reclustering, the RRCH cannot handle clusters with bad quality, such as overlay of clusters and too small or too big a cluster size.

2.2.1.5. CPEQ. CPEQ [8] adopts the CH selection scheme of LEACH. Instead of using the randomly selected node as a CH directly, CPEQ uses the randomly picked node to choose the node with the highest residue energy from its neighbors. To build clusters, CPEQ uses time-to-live (TTL) to limit the size of the cluster and calculates the optimized routes from cluster members to their CHs. For inter-cluster communication, CPEQ also uses the optimized multi-hop routes among CHs and the sink. By performing data aggregation within clusters and calculating optimized routes, CPEQ reduces traffic collision and data transmission delay.

In a large scale WSN, the flooding mechanism adopted by CPEQ in its initial stage may become problematic. Flooding incurs redundancy as a node sends data to its neighbor no matter if it already has it or needs it. Further, CPEQ is only appropriate for static and fixed networks due to the high cost of addressing all the nodes in the system, and hence the addresses are hard to maintain.

2.2.1.6. HEED. HEED [26] protocol is an energy-aware hierarchical approach improved from LEACH. HEED focuses on choose appropriate CHs by adding more network information. It uses residual energy as the primary clustering parameter to select a number of tentative CHs. Those tentative CHs inform their neighbors of their intentions to become CHs. These advertisement messages include a secondary cost measure that is a function of neighbor proximity or node degree. This secondary cost is used to guide the regular nodes in choosing the best cluster to join, and to avoid elected CHs being within the same range of each other. If a CH is far from the sink, it tries to send the aggregate data to another CH instead of sending to the sink directly.

2.2.2. Well-selected-CH protocols

To avoid the problem caused by random CH selection, there are many other approaches focusing on how to select appropriate CHs to achieve efficient communications.

2.2.2.1. LEACH-C. LEACH-C (LEACH-centralized) [15] is identical to the LEACH protocol as far as formatting clusters at the beginning of each round. However, instead of nodes randomly self-selecting as a CH, a centralized algorithm is performed by the sink in LEACH-C. The sink collects location information from the nodes, and then broadcasts its decision of which nodes are to act as CHs back to the nodes. The overall performance of LEACH-C is better than LEACH since it moves the duty of cluster formation to the sink. However, LEACH-C is sensitive to the sink location. Once the energy cost of communicating with the sink becomes higher than the energy cost for cluster formation, LEACH-C no longer provides good performance. Sinks may be located far from the network in most WSN applications. So, the dependence on the sink location is a major disadvantage of LEACH-C.

2.2.2.2. PEBECS. Focusing on the hot spot problem, PEBECS [23] presents the solution by dividing a WSN into several partitions with equal area and then grouping the nodes into unequally sized clusters. The shorter the distance between the partition and the sink, the more clusters are created within the partition. Further, to select the CH, PEBECS uses the node's residual energy, degree difference and relative location in network. PEBECS mitigates the hot spot problem by grouping nodes in smaller clusters to save more energy on their intra-cluster communication. As the result, PEBECS achieves longer network lifetime by better balancing node energy consumption.

2.2.2.3. MHP. Zhenghao et al. [29] proposed energy-efficient multi-hop polling (MHP) scheme to collect data from the two-layered heterogeneous sensor network. The carefully deployed cluster heads have more energy than the basic sensor nodes. In turns, each cluster head launches the discovering process to join the basic sensor nodes into its cluster. After the clusters are established, MHP minimizes the intra-cluster communication energy consumption by using polling to collect data from sensor nodes instead of the sensor nodes randomly reporting data. MHP presents a fast online polling algorithm to solve the problem of finding a contention-free polling schedule. However, MHP has stricter requirement of network deployment. The cluster head nodes have to be carefully deployed, otherwise, the part of network cannot be able to be functional. Further, MHP requires the knowledge of the sensor nodes' location.

2.2.2.4. DSC. Dynamic/Static Clustering protocol (DSC) [5] is an extension of LEACH-C. Using the scheme, each node gets its current location using a global positioning system (GPS) and sends the location information and energy status to the sink. The sink will then determine the number of CHs based on the collected information and broadcast the clustering result to each node. Each CH will also determine a TDMA scheme for its cluster members similar to LEACH. Compared with LEACH-C, the number of messages received at the sink for DSC is significantly reduced. However, it suffers similar problems that LEACH-C has.

2.2.2.5. EDASC. An energy-efficient data aggregation protocol based on static clustering (EDASC) [11] tries to reduce the overhead of dynamic clustering. The approach also adopts the LEACH model. But EDASC makes use of the sink to select an initiator to start the clustering process. The sink also broadcasts the CH schedule to sensor nodes. However, EDASC calculates the Hausdorff distance to determine CHs and it alternates the role of CH with an aim to prolong the network life. EDASC also has similar issue that LEACH-C encounters. The main idea of EDASC is to form clusters statically, which is similar to DHAC. Nevertheless, DHAC is fully distributed and does not rely on a centralized sink to start the cluster formation.

2.2.2.6. AHP. Analytical Hierarchy Process (AHP) [25] conducts CH selection algorithm by the sink. AHP supports mobile sensor nodes. Three factors are considered: energy, mobility, and the distance to the involved cluster centroid. AHP calculates local weight and global weight by using those three factors. AHP chooses the CHs by combining the results of these two weights.

To maintain the clusters, CH re-selection only occurs when selected CHs die or move to other clusters. Compared to LEACH, AHP improves the network lifetime based on the time of the last node dead [25]. Comparing to centralized protocols, AHP is more complex than LEACH-C, since AHP considers more factors. Because AHP needs to transmit more information from the network to the sink, the communication cost between nodes and the sink may cause much higher energy consumption.

2.2.2.7. EAD. EAD [7] presents an energy-aware algorithm to build a broadcast tree that spans all the sensor nodes with a maximum number of leaves. EAD turns off the radios of the leaf nodes and only use the non-leaf nodes to be in charge of data aggregation and relaying tasks. Further, EAD ensures that the leaf nodes save more energy without compromising the connectivity of the network. After each data-transmit phase, EAD will re-build the broadcast tree to identify all the dead nodes and orphaned nodes. EAD requires global knowledge of the network to build the optimized spanning tree, which causes higher constraints and more energy consumption.

The advantages and disadvantages of two kinds of hierarchical routing protocols are summarized in the following paragraphs.

- *Random-selected-CH protocols.* Although random-selected-CH protocols can bring more flexibility and toleration, these approaches have three main disadvantages. Firstly, the randomly picked CH may have a higher communication cost because it has no knowledge of intra-cluster or inter-cluster communication. Secondly, if periodic CH rotation is used to reduce the effect of CH random selection, the re-selection itself uses extra energy to re-build clusters. Periodic CH rotation also leads to an uneven wave of performance due to the non-stop change. Thirdly, the random selection cannot guarantee good protocol performance. In other words, the best arrangement and the worst arrangement have an equal chance to be used in the network.
- *Well-selected-CH protocols.* The well-selected-CH protocols can provide better cluster quality, but they usually have a more complex scheme and higher overhead to optimize the CH selection and cluster formation. Some approaches use the sink to help choose CHs by frequently collecting information from nodes. However, the sink performing the algorithm introduces another issue that increases communication cost between the nodes and the sink because they need to frequently exchange administrative information. Other researchers have to try to use the optimization algorithms to distinguish the roles of nodes. But there may not be enough fault tolerance in these schemes because any change to the network may cause the entire network to update information and perform reclustering.

3. Hierarchical agglomerative clustering (HAC) and its application to WSNs

HAC [3,21] algorithm is a conceptually and mathematically simple clustering approach to data analysis. It can provide very informative descriptions and visualization for the potential data clustering structures, especially when real hierarchical relationships exist in the data [24]. To apply the HAC algorithm in WSNs, we proposed six-step clustering to generate appropriate clusters.

3.1. Step 1: obtain the input data set

An input data set for HAC is a component–attribute data matrix. Components are the nodes that we want to group based on their similarities. Nodes exchange HELLO messages and obtain neighbor nodes' attributes. Attributes are the properties of the components such as the location of nodes, the RSS, the connectivity of nodes, or other features.

The type of input data set can be classified into quantitative data and qualitative data. Fig. 1 shows a randomly generated 8-node network in the $10 \times 10\ \mathrm{m}^2$ field. As illustrated in Table 1a, the location information is used as the quantitative input data. Table 1b uses the one-hop network connectivity data as the qualitative input data, where the "1" value represents a one-hop connection and the "0" value represents no direct connection.

3.2. Step 2: compute the resemblance coefficients

A resemblance coefficient for a given pair of nodes indicates the degree of similarity or dissimilarity between these two nodes. It could be quantitative (e.g., location, RSS) or qualitative (e.g., connectivity). We can calculate Euclidean distance based on the location information by using the Pythagorean Theorem. In Eq. (1), x and y represent the location of node, a and b, on x-axis and y-axis.

$$\text{Euclidean distance}: D_{ab} = [(a_x - b_x)^2 + (a_y - b_y)^2]^{1/2} \quad (1)$$

To deal with the qualitative data, there are various ways to calculate the resemblance coefficients [21]. There are three typical methods:

- JACCARD Coefficient:

$$C_{(a,b)} = N_{1-1}/(N_{1-1} + N_{1-0} + N_{0-1}) \quad (2)$$

- SORENSON Coefficient:

$$C_{(a,b)} = 2N_{1-1}/(2N_{1-1} + N_{1-0} + N_{0-1}) \quad (3)$$

- Simple Matching Coefficient:

$$C_{(a,b)} = (N_{1-1} + N_{1-0})/(N_{1-1} + N_{1-0} + N_{0-1} + N_{0-0}) \quad (4)$$

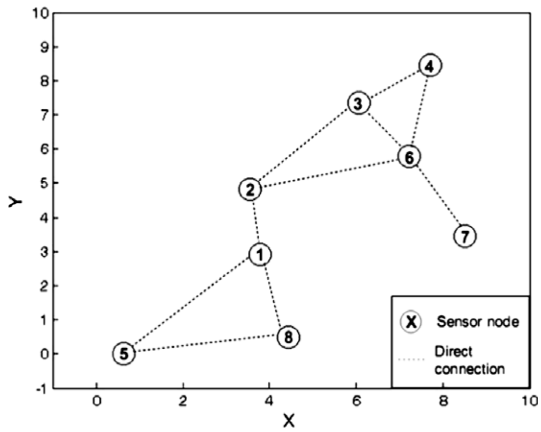


Fig. 1. A simple 8-node network.

Table 1
Node input data matrix for the 8-node network.

Component (node)	Attribute	
	x-Axis	*y*-Axis
(a) Quantitative data: node location data matrix		
{1}	3.78	2.9
{2}	3.56	4.83
{3}	6.06	7.34
{4}	7.71	8.46
{5}	0.63	0.01
{6}	7.23	5.78
{7}	8.52	3.46
{8}	4.43	0.48

	{1}	{2}	{3}	{4}	{5}	{6}	{7}	{8}
(b) Qualitative data: one-hop network connectivity data matrix								
{1}	1	1	0	0	1	0	0	1
{2}	1	1	1	0	0	1	0	0
{3}	0	1	1	1	0	1	0	0
{4}	0	0	1	1	0	1	0	0
{5}	1	0	0	0	1	0	0	1
{6}	0	1	1	1	0	1	1	0
{7}	0	0	0	0	0	1	1	0
{8}	1	0	0	0	1	0	0	1

N_{1-1}, N_{1-0}, N_{0-1}, N_{0-0} are counts of 1–1, 1–0, 0–1, and 0–0 matches of attribute pair between any two nodes, a and b. $C_{(a,b)}$ represents the value of the resemblance coefficient between the node a and node b. As an example, Table 2 presents the Euclidean distances between any of two nodes by using Eq. (1). Table 3 demonstrates use of a SORENSON dissimilarity coefficient into indicate the difference between any two nodes. Based on the data of Table 1b, the coefficients in Table 3 are calculated by using the SORENSON method, given by Eq. (3).

3.3. Step 3: execute the HAC algorithm

After building up the resemblance matrix, the HAC algorithm repeatedly identifies the minimal coefficient in the resemblance matrix and executes the clustering algorithm to assign the nodes into a "tree". Each step includes merging two clusters together and updating the resemblance matrix. Updating the resemblance matrix is an important step, and various algorithm methods can be adopted. There are four main types of the HAC algorithm methods [3].

Table 2
Resemblance matrix with quantitative data using Euclidean distance.

	{2}	{3}	{4}	{5}	{6}	{7}	{8}
{1}	1.94	4.99	6.81	4.27	4.49	4.77	2.51
{2}	–	3.54	5.51	5.64	3.79	5.15	4.44
{3}	–	–	1.99	9.12	1.95	4.59	7.05
{4}	–	–	–	11	2.72	5.07	8.63
{5}	–	–	–	–	8.77	8.61	3.83
{6}	–	–	–	–	–	2.65	5.99
{7}	–	–	–	–	–	–	5.06

- *Single LINKage (SLINK)*, also called the nearest neighbor method. It defines the similarity measure between two clusters as the minimum resemblance coefficient among all pair entities in the two clusters.

$$C_{\mathrm{SLINK}} = \mathrm{Min}(C_{(1,1)}, C_{(1,2)}, \cdots, C_{(i,j)}, \cdots, C_{(m,n)}) \tag{5}$$

- *Complete LINKage (CLINK)*, also called the furthest neighbor method. It defines the similarity measure between two clusters as the maximum resemblance coefficient among all pair entities in the two clusters.

$$C_{\mathrm{CLINK}} = \mathrm{Max}(C_{(1,1)}, C_{(1,2)}, \cdots, C_{(i,j)}, \cdots, C_{(m.n)}) \tag{6}$$

- *Un-weighted Pair-Group Method using arithmetic Averages (UPGMA)*. This defines the similarity measure between two clusters as the arithmetic average of resemblance coefficients among all pair entities in the two clusters. UPGMA is the most commonly adopted clustering method.

$$C_{\mathrm{UPGMA}} = \frac{1}{mn} \sum_{i=1,j=1}^{m,n} C_{(i,j)} \tag{7}$$

- *Weighted Pair-Group Method using arithmetic Averages (WPGMA)*. This is the simple arithmetic average of resemblance coefficients between two clusters without considering the cluster size.

$$C_{\mathrm{WPGMA}} = \frac{1}{mn} \sum_{i=1,j=1}^{m,n} W_i C_{(i,j)} \tag{8}$$

The results of the HAC algorithm are usually depicted by a binary tree or dendrogram, as shown in Fig. 2. The root node of the dendrogram represents the whole data set and each leaf is regarded as a node. The intermediate nodes, thus, describe the extent that the nodes are proximal to each other and the height of the dendrogram. Fig. 2 demonstrates three different clustering results for the simple 8-node network depicted in Fig. 1 by using SLINK, CLINK, UPGMA methods with quantitative data.

3.4. Step 4: cut the hierarchical cluster tree

To avoid clusters become oversized and to stop merging of clusters, we make a cut by using a pre-configured threshold value, such as transmission radius, number of clusters, or cluster density. Fig. 3b shows a cutting of transmission radius basing on the clustering result of UPGMA with quantitative data. Fig. 4 illustrates that the sample network has three corresponding clusters, {3, 6, 4, 7}, {1, 2, 8}, and {5} based on Fig. 3b.

Table 3
Resemblance matrix with qualitative data using SORENSON dissimilarity coefficients.

	{2}	{3}	{4}	{5}	{6}	{7}	{8}
{1}	0.5	0.75	1	0.143	0.778	1	0.143
{2}	–	0.25	0.429	0.714	0.333	0.667	0.714
{3}	–	–	0.143	1	0.111	0.667	1
{4}	–	–	–	1	0.25	0.6	1
{5}	–	–	–	–	1	1	0
{6}	–	–	–	–	–	0.429	1
{7}	–	–	–	–	–	–	1

3.5. Step 5: control the minimum cluster size

If the size of a cluster is smaller than the predefined threshold, minimum cluster size, the cluster merges with its closest neighboring cluster. Fig. 3c shows that the small Cluster {5} is merged into Clusters {1, 2, 8}. Thus, Fig. 5 presents the final formatted clusters {3, 6, 4, 7} and {1, 2, 8, 5}.

3.6. Step 6: choose CHs

Once clustering is finished, CHs can be initially determined by different strategies. In this paper, CHs are the nodes that satisfy two conditions: (i) the node is in the bottom level, which merged into the cluster in the first step and (ii) the node has the lower ID. As demonstrated in Fig. 5, nodes 1 and 6 are the CHs of the Cluster {1, 2, 8, 5} and {3, 6, 4, 7}, respectively.

4. Distributed hierarchical agglomerative clustering (DHAC)

In this paper, we propose a DHAC algorithm for distributed environments by tailoring the HAC algorithms. The main idea behind DHAC is that a node only needs one-hop neighbor knowledge to build clusters. To illustrate the feasibility of DHAC in WSNs, DHAC adopts the "general assumptions" of WSNs as follows:

- The nodes in the network are quasi-stationary.
- The nodes are left unattended after deployment.
- Each node only has local information or the identification of its one-hop neighbor nodes.
- All nodes have similar capabilities, processing, communication and initial energy.
- Propagation channel is symmetric.
- The transmission ranges of nodes are adjustable. All the nodes have the capability to communicate with the sink directly.
- The sink is static.

4.1. DHAC: cluster formation

Without the global knowledge, DHAC can make use of the neighboring information to determine if a node actually needs to perform the clustering task. The rationale is that every node knows its one-hop neighbors. Fig. 6 illustrates the pseudo code of the DHAC implementation for WSNs.

4.1.1. Step 1 and 2: obtain input data set and build resemblance matrix

The procedure "Set up ResM" (Fig. 6, lines 1–7) corresponds to "obtain input data set" and "build resemblance matrix" steps described in Section 3. To collect input data and set up the local resemblance matrix, in the beginning, each node elects itself as a cluster head and exchanges the information via HELLO messages with its neighbors. In

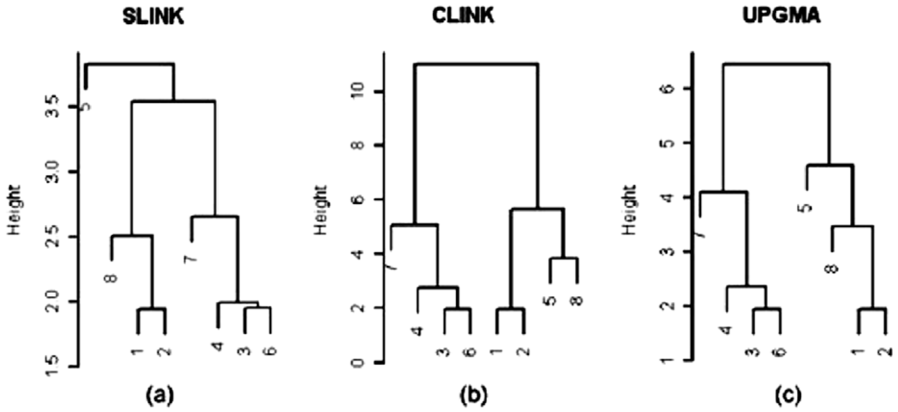


Fig. 2. Dendrogram using different the HAC algorithms with quantitative data.

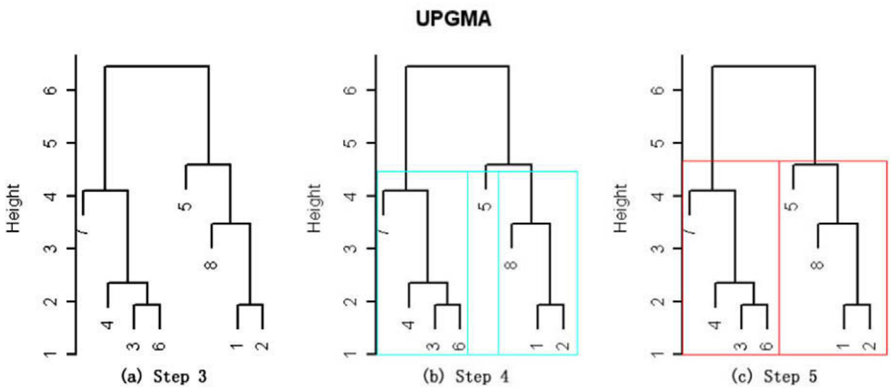


Fig. 3. Clustering steps and dendrogram using UPGMA with quantitative data.

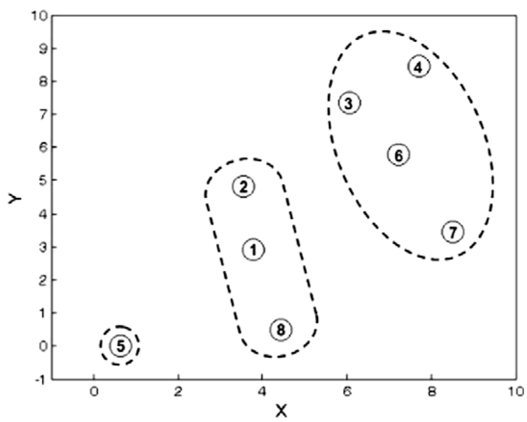


Fig. 4. Generated clusters at Step 4, using UPGMA with quantitative data.

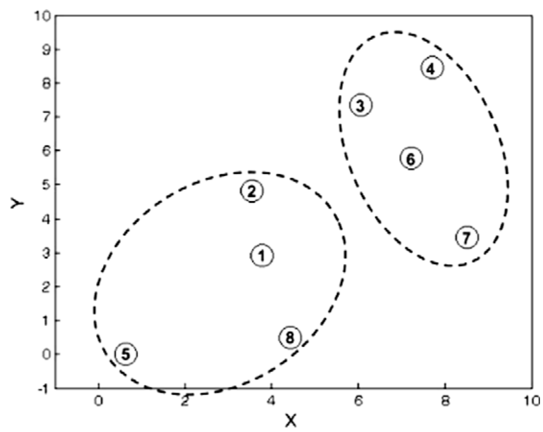


Fig. 5. Generated clusters at Step 5, using UPGMA with quantitative data.

Fig. 6, lines 1–7 initialize the clustering process by setting the current ID_{CH} as ID_{Node} and exchanging HELLO messages with one-hop neighbors.

To illustrate the algorithm, based on the network shown in Fig. 1, Table 4 presents the initial local resemblance matrices of the singleton clusters after exchanging HELLO messages. In Table 4, the resemblance coefficients are the quantitative Euclidean distance which is based on the location information from one-hop neighbors. For instance, at first node 1 itself forms an initial Cluster {1} and it has three neighbors, 2, 5, and 8.

4.1.2. Step 3: execute the DHAC algorithm

After the clustering process ends, each cluster establishes its own local resemblance matrix, from which its minimum coefficient (M_{Coeff}) can be easily found. Each cluster then determines its minimum cluster head (M_{CH}). If the ID_{Node} of the M_{CH} is larger than the CH, the CH will send an INVITE message to the M_{CH}. In Fig. 6, lines 10–16 specify two requirements: the sender node must be the current CH, and the sender ID_{Node} must be smaller than M_{CH}. Fig. 7 demonstrates each cluster's response based on its own resemblance matrix.

The following illustrates DHAC in detail, step by step:

- In Fig. 6, lines 17–23, when a CH receives an INVITE message, it checks the source of the message. If the source is its M_{CH}, the CH sends a CONFIRM message back to the source, elects the source to be the new CH, and turns into the sleep mode; otherwise, the CH sends back a REJECT message. For example, node 2 is the initial CH in Cluster {2}. When receiving an INVITE message from Cluster {1}, Cluster {2} sends a CONFIRM message back to Cluster {1}. In Fig. 7, Cluster {8} will send a REJECT message to Cluster {5} when it receives an INVITE message from Cluster {5}, since the M_{CH} of Cluster {8} is Cluster {1}.

Definitions:

ID_{Node}: Node ID

ID_{CH}: Cluster Head ID

ResM: Resemblance Matrix

M_{Coeff}: The minimum coefficient in the ResM

M_{CH}: The ID_{CH} of the cluster corresponds to the M_{Coeff} in the ResM

C_{size}: The number of cluster member in a cluster

1. **procedure** Set Up ResM   > *Exchange information with neighbors*
2. $ID_{CH} \leftarrow ID_{Node}$.   > *Self forms a singleton cluster*
3.   Send HELLO message to one-hop neighbors
4.   ....   > *Keep listening to neighbors*
5.   Receive HELLO messages
6.   Establish ResM
7. **end procedure**

8. **procedure** Execute DHAC
9.   **while** M_{Coeff} < Threshold **do**
10.   **if** $ID_{CH} == ID_{Node}$ **then** .   > *Sending message*
11. $M_{Coeff} \leftarrow$ minimum value in ResM
12. $M_{CH} \leftarrow ID_{CH}$ with M_{Coeff}
13.   **if** $ID_{CH} < M_{CH}$ **then**
14.   Send INVITE message to M_{CH}
15.   **end if**
16.   **end if**

....   > *Receiving message*

17.   **if** Received INVITE message **then**
18.   **if** M_{CH} = Sender ID_{CH} **then**

Fig. 6. Pseudo code of DHAC in WSNs.

```
19.                 send CONFIRM message
20.                 ID_CH ← Sender ID_CH
21.             else
22.                 Send REJECT message
23.             end if
24.         else if Received CONFIRM message then GoTo MERGE CLUSTERS
25.         else if Received INFORM message then Update ResM
26.         else if Received REJECT message then Stop sending INVITE message
                                                  until ResM updated
27.         end if
28.     end while
29.     if C_size < Minimum Cluster Size then
30.         GoTo MERGE CLUSTERS
31.     end if
32. end procedure

33. procedure MERGE CLUSTERS .                    > Cluster merging procedure
34.     Merge two clusters
35.     Combine neighboring information of two clusters
36.     Update ResM by using chosen method
37.     Broadcast INFORM message
38. end procedure
```

Fig. 6 (*continued*)

- Once a CH receives a REJECT message from its M_{CH}, the CH stops sending the INVITE message until its resemblance matrix has been updated. For example, Cluster {5} will stop sending the invite message to Cluster {8}.
- After receiving a CONFIRM message, the CH updates its neighbor list and resemblance matrix, then it merges with the cluster headed by M_{CH} to form a new cluster (Fig. 6, line 33–38). The resemblance matrices of two clusters are updated through the chosen HAC algorithm. Clusters {1} and {2}, for instance, will merge their resemblance matrices and their neighbor lists.
- The CH of the new cluster broadcasts an INFORM message to notify its neighbors to update their resemblance matrices (Fig. 6, lines 25 and 37). Clusters update their own resemblance matrix after receiving this INFORM message, which contains the new cluster information and the merged neighbor list.

The process will repeat until the while condition (line 9) fails. During the clustering process, all CHs of clusters keep listening. When a CH receives a message, it reacts based on the message type. Table 5 lists the updated resemblance matrices of new clusters using SLINK with quantitative data after the first round of the DHAC algorithm execution.

4.1.3. Step 4: cut the hierarchical cluster tree

Using a predefined threshold, the while loops in Fig. 6, lines 9–28, controls the upper bound size of clusters. The control conditions correspond to the step of cutting the hierarchical cluster tree.

4.1.4. Step 5: control the minimum cluster size

After clusters are generated by running DHAC, the minimum cluster size can also be used to limit the lower bound of cluster size by performing the procedure "MERGE CLUSTERS" (Fig. 6, lines 29–31).

4.1.5. Step 6: choose CHs

To choose the corresponding CHs, DHAC simply choose the lower ID node between the two nodes that join the cluster at the first step. The CH chosen does not require extra processing.

Table 4
Initial local resemblance matrices with quantitative data using Euclidean distance.

Neighbor	Coefficient
Cluster {1}	
{2}	1.94
{5}	4.27
{8}	2.51
Cluster {2}	
{1}	1.94
{3}	3.54
{6}	3.79
Cluster {3}	
{2}	3.54
{4}	1.99
{6}	1.95
Cluster {4}	
{3}	1.99
{6}	2.92
Cluster {5}	
{1}	4.27
{8}	3.83
Cluster {6}	
{2}	2.79
{3}	1.95
{4}	2.72
{7}	2.65
Cluster {7}	
{6}	2.65
Cluster {8}	
{1}	2.51
{5}	3.83

4.1.6. Evaluation of cluster formation

Section 3 presents various options or variants of the HAC algorithm. We have conducted detailed investigation of DHAC and evaluated the effect of several parameters and methods that are critical to DHAC. The investigation in the area of cluster formation includes cluster size, cluster range, and clustering methods. We used different topologies with 100 nodes, uniformly distributed in a $100 \times 100\,\text{m}^2$ field with different node degrees. To ensure fair comparison of various HAC methods and reduce random effects caused by the simulation, we execute 30 simulation runs for each scenario and calculate the mean results. Detailed comparisons can be found in [30]. The next paragraph highlights the results of some common HAC algorithms.

With quantitative data, SLINK and CLINK result in two extreme cases. SLINK tends to produce compact cluster and the size of the clusters generated by SLINK is widely spread, ranging from 2 to 98 when the average node degree is below 5. Unbalanced cluster size can cause some nodes to die quickly, while some nodes cannot report data in time due to longer delays in scheduling. On the other hand, CLINK produces many clusters. More clusters provide shorter network reaction time at the expense of heavy inter-cluster communications. Compared to SLINK and CLINK, UPGMA and WPGMA generate similar and more balanced numbers of clusters. Cluster range is another important criterion to evaluate clustering methods. The cluster range is defined as the maximum distance between any two cluster members. In terms of cluster range, based on the experimental results, SLINK generates the largest range while UPGMA produces the smallest cluster range. Hence, UPGMA is used for more simulation comparisons that are shown in Section 5.

When DHAC used with qualitative data, a comparison of different resemblance coefficient calculation methods with the quantitative location data was conducted for evaluation. Many methods are available to calculate the resemblance coefficients. There are three well-known methods, JACCARD, SORENSON and simple matching, as illustrated in Section 3. The UPGMA method was used due to the better results it produced for quantitative data. The cluster range drops with different rates when the node degree increases. All three methods using qualitative data generate a larger maximum cluster range than when using quantitative location data. The simple matching method generates the largest cluster range. When the node degree increases, the cluster range of simple matching method remains almost unchanged while other methods decrease. The result obtained from the SORENSON calculation is closer to that of quantitative location data than other methods. Therefore, the SORENSON method is selected for further studies and comparisons shown in Section 5.

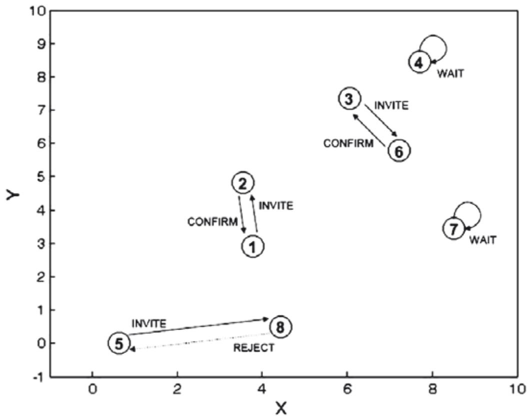
4.2. DHAC: cluster maintenance

After the cluster formation phase, DHAC uses the sequence of nodes merging into the current cluster as the schedule. Each cluster member gets its assigned role and starts to send data to CH in turns. DHAC uses the "TDMA + CDMA" model as the MAC layer protocol structure. During the cluster maintenance phase, DHAC adopts the TDMA frame for intra-cluster communications. For inter-cluster communication, we adopted an assumption: all the nodes have the capability to communicate with the sink directly, which is from the model used for LEACH and many other similar works. In the following discussion, DHAC uses CDMA to avoid the collision of communications among different clusters and makes the chosen CHs to communicate with the sink directly without using any sophisticated routing strategy. In DHAC, we consider the CHs as the representatives of their member nodes because the CHs collect data from their members and the members do not communicate outside of the cluster with CDMA. With its hierarchical structure, DHAC can flexibly choose a flat-routing protocol to get more efficient intra-cluster communications. Since many research efforts have focused on solving the routing issue in sensor networks with different application scenarios, this paper emphasizes on the cluster establishment and intra-cluster communications.

Since all the responsibilities of the CH require high energy dissipation, the CH role has to be rotated to different nodes. DHAC uses automatic CH rotation and re-scheduling to make energy dissipate uniformly through the whole network.

4.2.1. Automatic CH rotation

Automatic CH rotation uses time to control the CH rotation. After the cluster has run for several frames, the CH



Cluster {1}	Sends an INVITE message to Cluster {2} since the M_{Coeff} in Cluster {1} corresponds to Cluster {2} and the ID_{node} of CH in Cluster {2} is larger than CH in Cluster {1};
Cluster {2}	Expects an INVITE message from Cluster {1} since the ID_{node} of CH in Cluster {2} is larger than the CH in Cluster {1};
Cluster {3}	Sends an INVITE message to Cluster {6}, similar to Cluster {1}.
Cluster {4}	Expects an INVITE message from Cluster {3}. CH of Cluster {4} has a larger ID_{node} than the CH of Cluster {3}, and therefore will not send any INVITE message to Cluster {3};
Cluster {5}	Sends an INVITE message to cluster {8}, similar to Cluster {1}.
Cluster {6}	Expects an INVITE message from Cluster {3}, similar to Cluster {2}.
Cluster {7}	Has only one neighbor, similar to Cluster {4}.
Cluster {8}	Expects an INVITE message from Cluster {1} yet received an INVITE message from Cluster {5}. Cluster {8} will send a REJECT message to Cluster {5}.

Fig. 7. The tasks of clusters based on its own local resemblance matrix.

Table 5
The first round of DHAC: updated resemblance matrices using SLINK with quantitative data.

Neighbor	Coefficient
Clusters {1,2}	
{5}	4.27
{8}	2.51
{3, 6}	3.54
Member 2	1.94
Clusters {3,6}	
{1,2}	3.54
{4}	1.99
{7}	2.65
Member 6	1.95
Clusters {4}	
{3,6}	1.99
Clusters {5}	
{1,2}	4.27
{8}	3.83
Clusters {7}	
{3,6}	2.65
Clusters {8}	
{1,2}	2.51
{5}	3.83

role automatically passes to the next node in the schedule. When the CH role moves, every node still follows the same time slot, the previous CH changes to sleep mode at the end of its time slot and the new CH wakes up at the appropriate time and starts to collect data from other cluster members. This CH rotation does not need any message exchange because every node of the cluster already has the schedule from when the cluster formatted. In other words, automatic CH rotation does not consume extra energy.

4.2.2. Re-scheduling

Re-scheduling is designed to keep energy dissipation more uniform within the cluster. Under certain conditions, the CH informs cluster members of a new schedule, then, cluster members start to follow the new schedule. There are two conditions that can trigger the re-scheduling.

- A cluster has a change. The change can be a new node joining, or a node leaving or dead.
- The CH has lower energy than the threshold Th(energy).

$$\mathrm{Th}(\mathrm{energy}) = P(\mathrm{energy}) \times \frac{1}{n}\sum_{i=1}^{n} R(\mathrm{energy})_i \tag{9}$$

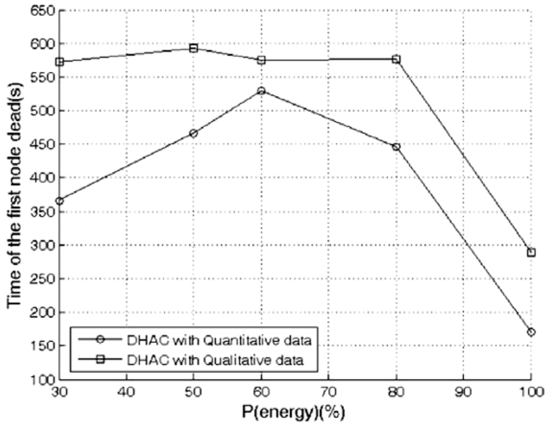


Fig. 8. Network lifetime of the first node dead versus P(energy) (%).

When cluster members send data to the CH, they also send their current residual energy. Based on the collected information, CH can calculate the average residual energy, R(energy), within the cluster. Once the R(energy) of the CH becomes lower than the Th(energy), the CH informs every cluster member of the new schedule. The new schedule is the sequence of the nodes in order of decreasing R(energy). Eq. (9) presents the calculation of the threshold Th(energy) by using a predefined percentage of average residual energy, P(energy).

4.2.3. Evaluation of cluster maintenance

Due to the re-scheduling energy cost, DHAC needs to carefully choose the threshold, Th(energy), to avoid frequent re-scheduling while achieving more uniform energy dissipation. In other words, more uniform energy dissipation and reduced energy dissipation can both be critical to the lifetime of network. Thus, the network lifetime is used to help determine the Th(energy). In Eq. (9), DHAC needs to specify the percentage of average residual energy, P(energy), to calculate Th(energy). A number of simulation experiments have been conducted to determine the value of P(energy).

Table 6
Simulation parameters.

Parameter	Value
Node number	100
Sensing filed range	(0, 0)–(100, 100)
Channel bandwidth	1 Mbps
Threshold distance (d_0)	5 m
E_{elec}	50 nJ/bit
$\bar{C}_{fs}$	10 pJ/bit/m^2
$\bar{C}_{mp}$	0.0013 pJ/bit/m^4
E_{fusion}	5 nJ/bit/signal
Data rate	5 TDMA frames per 10 s
$E_{initial}$	1 J/node
Exchanged message size	
Data packet size	500 bytes
HELLO	25 bytes
INVITE	25 bytes
SCHEDULE	25 bytes
CONFIRM	225 bytes

As depicted in Fig. 8, when P(energy) is changed from 30% to 80%, the DHAC with qualitative data maintain almost the same performance, whereas the DHAC with quantitative data suffers a large variation, reaching the best performance at P(energy) = 60%. Once P(energy) is larger than 80%, the performance of both input data types quickly deteriorates. With both quantitative and qualitative data, P(energy) = 60% offers good balance between re-scheduling energy cost and uniform energy dissipation. Therefore, P(energy) is fixed at 60% for subsequent simulation analyses.

5. Simulation and performance results

This section presents the performance comparison among the proposed DHAC, LEACH, and LEACH-C protocols. As presented in Section 2, LEACH is a typical random-selected-CH protocol and LEACH-C is a well-selected-CH protocol; both are important hierarchical routing protocols. Simulation experiments are carried out in the network simulator NS-2 (version ns-2.29). To minimize the influence of random network deployment, protocols are executed in 10 different network topologies and the mean results are used for comparison.

The simulated WSNs consist of 100 homogeneous sensor nodes randomly deployed within the sensing field from (0, 0) to (100, 100). Simulation parameters are shown in Table 6. The model and most of the parameters are similar to those in [15]. Using these parameters, a simple energy dissipation model of radio and processor hardware is presented as follows.

- *Receiving energy dissipation*. The energy consumed by receiving 1 bit data, E_{elec}, depends on coding and modulation. The energy consumed by receiving an L bit message is thus given by

$$E_{Rx} = L \times E_{elec} \qquad (10)$$

- *Transmitting energy dissipation*. Both the free space and the multi-path fading channel models are used [20,9]. When the distance between the transmitter and receiver, D, is larger than a specified threshold distance, d_0, the channel switches to the multi-path fading model,

$$E_{Tx} = L \times E_{elec} + L \times \varepsilon_{fs} \times D^4 \qquad (11)$$

Otherwise the channel follows the free space model ([14], p. 84),

$$E_{Tx} = L \times E_{elec} + L \times \varepsilon_{mp} \times D^2 \qquad (12)$$

- *Computation energy dissipation*. Data aggregation and resemblance matrix updating cause computation energy dissipation,

$$E_{com} = E_{fusion} \times \mathrm{Size}_{Signal} \times \mathrm{Number}_{Signal} \qquad (13)$$

The equation defines the computational costs of performing data calculation [[14], p. 86]. To perform data aggregation, CHs compress the collected data by using the computation energy method depicted in Eq. (13). Thus, in our simulation, the CHs will consume certain amount of

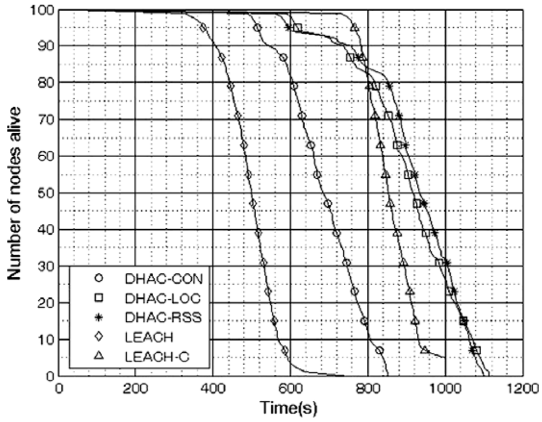


Fig. 9. Network lifetime, the sink at (50, 50).

energy, E_{com}, to calculate collected data and compress the data into one data packet.

We use three metrics to analyze and compare our simulation results for clustering and energy saving: network lifetime, energy dissipation and the number of data packets received at the sink. We use $T_{n\%}$ to represent the network lifetime when there are $n\%$ nodes dead. The metrics are evaluated for LEACH, LEACH-C and DHAC with different input data types. *DHAC-LOC* is the DHAC with location quantitative data, *DHAC-RSS* is DHAC with the RSS quantitative data and *DHAC-CON* is the DHAC with the connectivity qualitative data.

Fig. 9 presents the network lifetime corresponding to different protocols when the sink locates at the center of network. It can be observed that DHAC-LOC and DHAC-RSS have the longest $T_{100\%}$. LEACH has the shortest network lifetime: the $T_{1\%}$ is 364 s and only 10% of the nodes are alive at 585 s. Compared to LEACH, DHAC-CON prolongs $T_{1\%}$ by 13.5%. Compared to DHAC, LEACH-C provides a longer lifetime than DHAC-CON but a shorter lifetime than DHAC-LOC and DHAC-RSS. Although LEACH-C has a smaller death rate at the beginning of network serving (until around 15% of the nodes are dead), DHAC-LOC and DHAC-RSS have more nodes alive the rest of the time which means better sensing coverage. DHAC-LOC and DHAC-RSS provide around 100 s longer $T_{100\%}$ than LEACH-C.

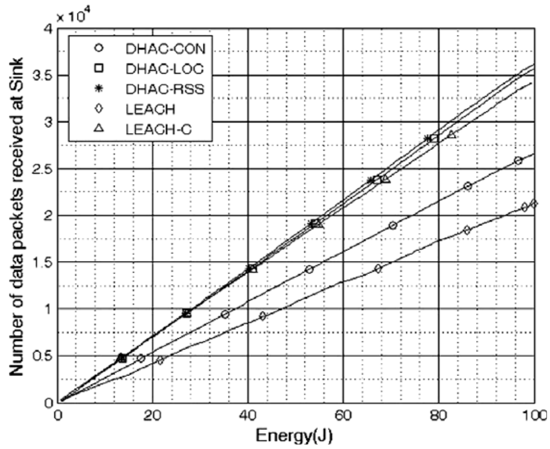


Fig. 11. Total energy dissipation versus the amount of data packets received to the sink (50, 50).

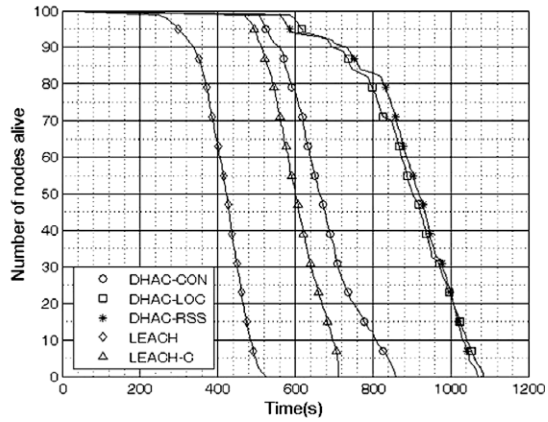


Fig. 12. Network lifetime, the sink at (50, 300).

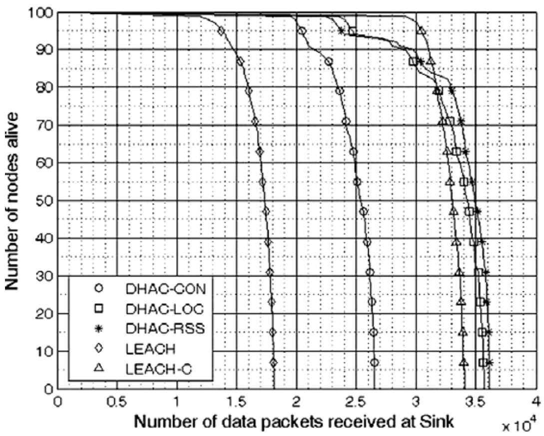


Fig. 10. Number of nodes alive versus the amount of data packets received at the sink (50, 50).

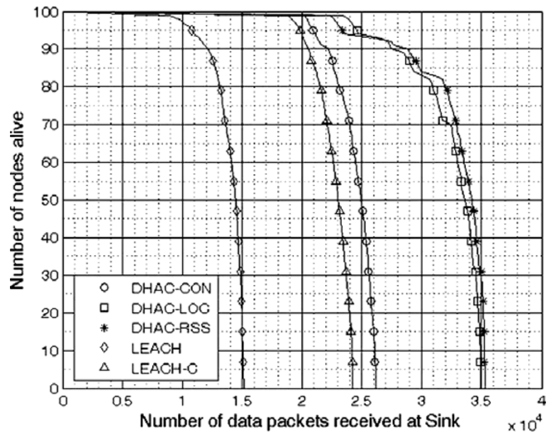


Fig. 13. Number of nodes alive versus the amount of data packets received at the sink (50, 300).

Fig. 10 shows the number of data packets received at the sink. Similar to what is observed in Fig. 9, DHAC-LOC and DHAC-RSS outperform LEACH and slightly outperform LEACH-C. When the last node died, the sink in DHAC-LOC and DHAC-RSS received approximately 36,000 data packets, and DHAC-CON reached 26,500 data packets; LEACH received 18,000 packets and LEACH-C received 34,000 packets.

To further investigate energy efficiency, we examined the relationship between total energy dissipation and the amount of data packets received at the sink. As illustrated in Fig. 11, higher slopes signify higher efficiency since

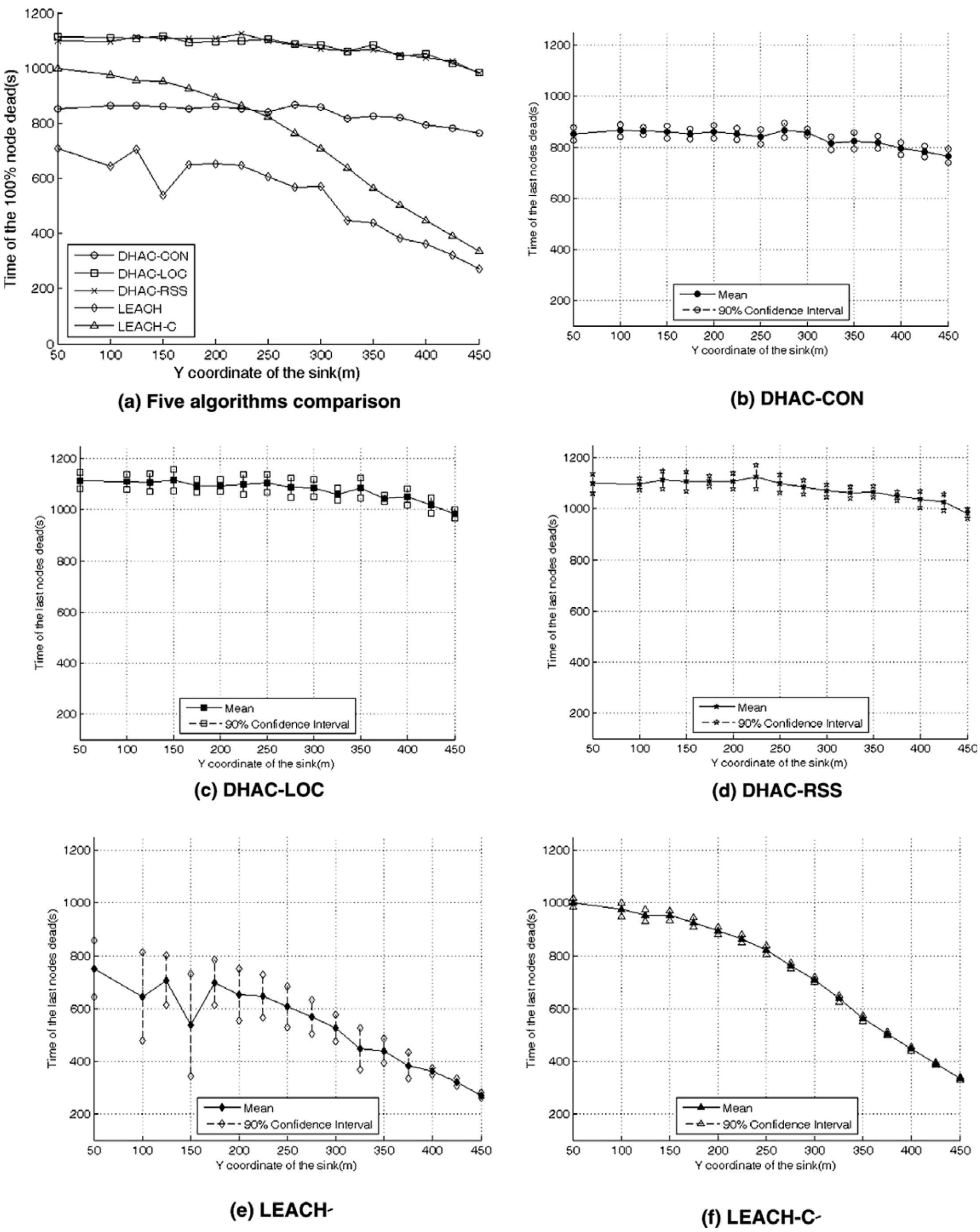


Fig. 14. Time of the last nodes dead against the sink location.

nodes send out more data with less energy. DHAC-LOC and DHAC-RSS offer the best energy efficiency, while LEACH provides the worst efficiency. LEACH-C still performs better than DHAC-CON and slightly worse than DHAC-LOC and DHAC-RSS.

In most applications of WSNs, the sink is usually deployed far from the network field. Network nodes spend more energy to communicate with the sink when it is located far from the network. Fig. 12 shows the performance of LEACH-C quickly decreases because a centralized protocol has to depend on the communication between the network and the sink. Thus, LEACH-C requires another constraint with the location and capability of the sink. In Fig. 12, DHAC provided longer network lifetime than either LEACH or LEACH-C. The $T_{100\%}$ of LEACH-C is reduced to 709 s from 1000 s when the sink is located at (50, 50). In Fig. 12, LEACH-C has 29% reduction of the amount of data packets received at the sink than previous experiments, while other protocols only have slight reductions in the data received at the sink (see Fig. 13).

Moving the sink might cause huge changes of energy dissipation. Fig. 14 investigates the effects of sink locations by moving the sink from (50, 50) to (50, 450) while fixing the X coordinate as 50. In Fig. 14a, while the sink is moving further away, DHAC-LOC and DHAC-RSS keep approximately a factor of two times the lifetime compared with LEACH, and DHAC-CON has a factor of 1.5 improvements. While the sink moves further, the network lifetime of LEACH-C decreases very quickly. After the sink moves further than (50, 250), the performance of LEACH-C becomes lower than DHAC. The performance of LEACH-C quickly decreases because a centralized protocol has to rely on the communication between the network and the sink. As a result, LEACH-C can not suit many applications because the sink is usually deployed far from the network field.

Fig. 14b–f individually indicates the $T_{100\%}$ against the location of the sink by using the confidence interval. Solid marks indicate sample mean values μ, and hollow marks show two-sided 95% confidence interval, $\mu \pm \delta$, based on 10 different topologies. Since each sample mean μ only used 10 samples and we have no *a priori* knowledge on the standard deviation of the population, the confidence interval is related to Student's t-distribution rather than Gaussian distribution. Therefore δ is given by $t_{(1-\alpha)/2,\nu}\frac{\sigma}{\sqrt{N_s}}$, where $N_s = 10$ is the number of samples, $\sigma = \sqrt{\frac{\sum_{i=1}^{N_s}(T_i-\mu)^2}{n-1}}$ is the estimated standard deviation of the population, and $t_{(1-\alpha)/2,\nu} = t_{0.05,9} = 2.262$ is a two-sided critical value for a t-distribution (in our case, the confidence level $\alpha = 95\%$ and the freedom of t-distribution $\nu = n - 1 = 9$). Thus, the two-sided 95% confidence interval is $\mu \pm 2.262\frac{\sigma}{\sqrt{10}}$. Note that the population distribution of "Time of % nodes dead" is still assumed to be Gaussian. Refer to [[6], p. 146–155] for details. Given the fixed confidence level $\alpha = 95\%$ Fig. 14e shows LEACH has the widest confidence intervals due to its randomness on CH selection, while LEACH-C, shown in Fig. 14f, has the narrowest confidence intervals than others. When the sink moves, DHAC provides more stable performance than LEACH and LEACH-C.

Fig. 15 depicts that the DHACs gain much better performance when the network has light traffic. The data rate is changed from 1 frame per 10 s to the maximum of 15 frames per 10 s. DHAC-LOC and DHAC-RSS have approximately 2500 s longer $T_{100\%}$ than LEACH when the data rate decreases to only 1 frame per 10 s. When the data rate reaches 15 frames per 10 s, DHACs still have 145 s (from 236 s to 381 s) longer $T_{100\%}$ than LEACH and LEACH-C. Since LEACH and LEACH-C use a fixed period to control CH rotation, the reclustering does not consider whether the CH has enough energy to coordinate its cluster. When energy dissipation is not fast and residual energy of CH is enough, DHAC consumes less energy on re-scheduling than LEACH and LEACH-C, which reclustering after a fixed period of time. As a result, DHACs gain much better performance when the network has light traffic.

To examine the influence of network topology, we change the number of nodes from 50 to 250 within 100×100 m^2 field and observe the network lifetime, $T_{100\%}$, as shown in Fig. 16. When the node number is 50, all algorithms have much shorter network lifetime since each node has to spend more energy to communicate with other nodes and manage the cluster. And the network lifetime increases with the scale of network: the $T_{100\%}$ of

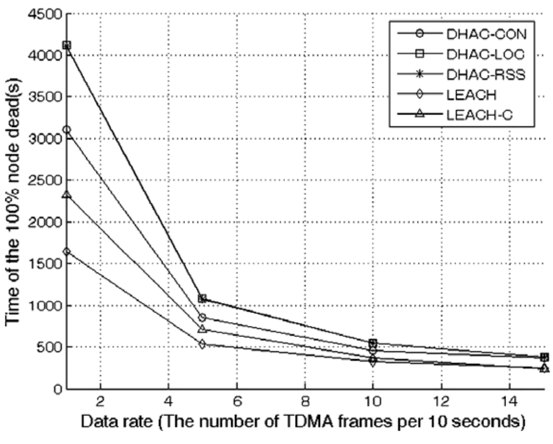


Fig. 15. Time of the 100% nodes dead versus different data rate.

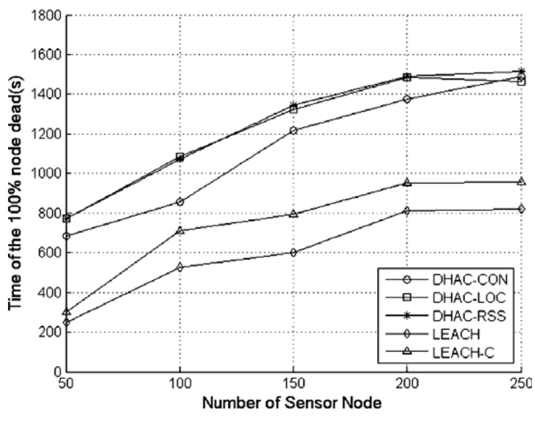


Fig. 16. Time of the 100% nodes dead versus number of sensor nodes.

DHACs' is improved 748 s on average when the node number is changed from 50 to 250. Meantime, the network life of DHAC-CON approaches DHAC-LOC and DHAC-RSS, because DHAC-CON can result in more balanced clusters as more neighbor information can be obtained. As we can observe from Fig. 16, compared with LEACH and LEACH-C, DHACs at least provide 383 s (from 301 s to 684 s) longer $T_{100\%}$ when there are 50 nodes, and 505 s (from 959 s to 1464 s) when there are 250 nodes. Thus, we expect the better performance from DHAC in large-scale networks than that of LEACH and LEACH-C.

6. Conclusions

To adapt to the constraints of WSNs, many hierarchical routing protocols have been proposed with different design goals, clustering criteria and basic assumptions. This paper advocated the application of well-known the HAC algorithm to WSNs and proposed a distributed approach, DHAC, to classify sensor nodes into appropriate groups instead of simply gathering nodes to some randomly selected CHs. We demonstrated the application and evaluation of methods, SLINK, CLINK, UPGMA, and WPGAM, with quantitative and qualitative data. We illustrated how to use the DHAC approach to mitigate the problems encountered with current protocols.

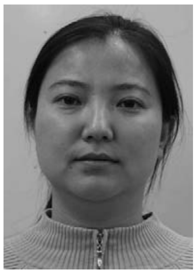
Our simulation model uses the networks with a number of nodes uniformly distributed in a $100 \times 100\,m^2$ square field. DHAC is directly compared to two well-known protocols, LEACH and LEACH-C, with four criteria, network lifetime, energy dissipation, number of data packets received at the sink, and network density. DHAC could also be indirectly compared against other clustering approaches that have been compared with LEACH. Various scenarios that use different sink locations, data rates and total number of nodes have been evaluated. The results demonstrate that DHAC outperforms LEACH in all criteria. Compared to LEACH-C, DHAC with quantitative data has better performance in all criteria, and DHAC with qualitative data performs better than LEACH-C when the sink is located far from the network area. When the number of node increases, DHACs still show higher performance than that of LEACH and LEACH-C. All the simulation results reveal that DHACs provide higher energy efficiency to meet the constraints of WSNs. In addition, DHACs are more flexible in terms of different WSN application scenarios. For instance, if GPS is not available, we can use either RSS or connectivity information to support clustering. Further, the performance of DHAC-RSS is very close to that of DHAC-LOC for various scenarios. RSS can be used to estimate the distance between nodes to support heterogeneous sensor devices or if the GPS component of some nodes fails. DHAC-RSS can even replace DHAC-LOC to reduce the cost for GPS.

References

[1] K. Akkaya, M. Younis, A survey on routing protocols for wireless sensor networks, Ad Hoc Networks 3 (3) (2005) 325–349.
[2] J.N. Al-Karaki, A.E. Kamal, Routing techniques in wireless sensor networks: a survey, IEEE Wireless Commun. 11 (6) (2004) 6–28.
[3] M.R. Anderberg, Cluster Analysis for Applications, Academic Press Inc., New York, 1973.
[4] T. Anker, D. Bickson, D. Dolev, B. Hod, Efficient clustering for improving network performance in wireless sensor networks, in: Proceedings of the 5th European Conference on Wireless Sensor Networks, March, 2008.
[5] F. Bajaber, I. Awan, Dynamic/static clustering protocol for wireless sensor network, in: Proceedings of the 2nd European Symposium on Computer Modeling and Simulation, September 2008, pp. 524–529.
[6] S. Bernstein, R. Bernstein, Schaum's Outline of Elements of Statistics I: Descriptive Statistics and Probability, McGraw-Hill Professional, 1999.
[7] A. Boukerche, X. Cheng, J. Linus, A performance evaluation of a novel energy-aware data-centric routing algorithm in wireless sensor networks, Wireless Networks 11 (5) (2005) 619–635.
[8] A. Boukerche, R.W. Pazzi, R. B Araujo, Fault-tolerant wireless sensor network routing protocols for the supervision of context-aware physical environments, J. Parall. Distr. Comput. 66 (4) (2006) 586–599.
[9] J. Broch, D. Maltz, D. Johnson, Y. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, in: Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking, 1998, pp. 85–97.
[10] H. Chen, C.S. Wu, Y.S. Chu, C.C. Cheng, L.K. Tsai, Energy residue aware (ERA) clustering algorithm for leach-based wireless sensor networks, in: 2nd International Conference ICSNC, Cap Esterel, French Riviera, France, August 2007, p. 40.
[11] S.-G. Den, L.-F. Shen, X.-R. Zhu, Energy-efficient data aggregation protocol based on static clustering for sensor networks, in: Proceedings of Progress in Electromagnetics Research Symposium, Hangzhou, China, March, 2008, pp. 470–473.
[12] W.R. Heinzelman, J. Kulik, H. Balakrishnan, Adaptive protocols for information dissemination in wireless sensor networks, in: Proceedings of ACM/IEEE 5th International Conference MobiCom, Seattle, Washington, USA, August 1999, pp. 174–185.
[13] W.B. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of 33rd Hawaii International Conference on System Sciences (HICSS), Wailea Maui, Hawaii, USA, January 2000, p. 10.
[14] W.B. Heinzelman, Application-specific protocol architectures for wireless networks, Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, USA, 2000.
[15] W.B. Heinzelman, A. Chandrakasan, H. Balakrishnan, An application-specific protocol architecture for wireless microsensor networks, IEEE Transactions on Wireless Communications 1 (4) (2002) 660–670.
[16] C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Proceedings of ACM/IEEE 6th International Conference on MobiCom, Boston, USA, August 2000, pp. 56–67.
[17] C.-H. Lung, C. Zhou, Y. Yang, Applying hierarchical agglomerative clustering to wireless sensor network, in: Proceedings of International Workshop on Theoretical and Algorithmic Aspects of Sensor and Ad-hoc Networks, June 2007, pp. 97–105.
[18] A. Martirosyan, B. Azzedine, R.W. Pazzi, A taxonomy of cluster-based routing protocols for wireless sensor networks, in: Proceedings of the International Symposium on Parallel Architectures, Algorithms, and Networks, 2008, pp. 247–253.
[19] D.-H. Nam, H.-K. Min, An efficient ad-hoc routing using a hybrid clustering method in a wireless sensor network, in: Proceedings of IEEE 3rd International Conference of WiMOB, White Plains, New York, USA, October 2007, p. 60.
[20] T. Rappaport, Wireless Communications: Principles and Practice, IEEE Press, Piscataway, NJ, USA, 1996.
[21] H.C. Romesburg, Cluster Analysis for Researchers, Krieger Publishing Company, Malabar, Florida, 1990.
[22] TinyOS: <http://www.tinyos.net/>.
[23] Y. Wang, T.L.X. Yang, D. Zhang, An energy efficient and balance hierarchical unequal clustering algorithm for large scale sensor network, Inform. Technol. J. 8 (1) (2009) 28–38.
[24] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Networks 16 (3) (2005) 645–678.
[25] Y. Yin, J. Shi, Y. Li, P. Zhang, Cluster head selection using analytical hierarchy process for wireless sensor networks, in: Proceedings of IEEE 17th International Symposium PIMRC, Helsinki, Finland, 2006, pp. 1–5.
[26] O. Younis, S. Fahmy, Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach, in: Proceedings of IEEE INFOCOM, Hong Kong, China, 2004, pp. 629–640.

[27] O. Younis, M. Krunz, S. Ramasubramanian, Node clustering in wireless sensor networks: recent developments and deployment challenges, IEEE Network 20 (3) (2006) 20–25.
[28] J.Y. Yu, P.H.J. Chong, A survey of clustering schemes for mobile ad hoc networks, IEEE Commun. Surv. Tutorials 7 (1) (2005) 32–48.
[29] Z. Zhang, M. Ma, Y. Yang, Energy-efficient multihop polling in clusters of two-layered heterogeneous sensor networks, IEEE Trans. Comput. 57 (2) (2008) 231–245.
[30] C. Zhou, C.-H. Lung, Application and evaluation of hierarchical agglomerative clustering in wireless sensor networks, in: Makki et al. (Eds.), Sensor and Ad-hoc Networks: Theoretical and Algorithmic Aspects, Lecture Notes in Electrical Engineering, Springer, 2008, pp. 255–276.

Chung-Horng Lung received his B.Eng. degree in Computer Science and Engineering from Chung-Yuan University, Taiwan, in 1983, and the M.S. and Ph.D. degrees in Computer Science and Engineering from Arizona State University in 1988 and 1994, respectively. In September 2001, he joined the Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada, where he is now an associate professor. He was with Nortel Networks from 1995 to 2001. At Nortel, he worked in the Software Engineering Analysis Lab (SEAL) as a senior software engineer and internal consultant, and Optical Packet Interworking (OPi) as a senior software designer. He worked in the Electronics Research and Service Organization (ERSO), Taiwan, in 1985. His research interests include: Wireless Ad Hoc and Sensor Networks, Communication Networks, Software Engineering, Distributed & Parallel Computing, and Network-Based Control Systems.



Chenjuan Zhou received her B.Eng. degree in Telecommunication Engineering from Yunnan University, China, in 2000, and her M.A.Sc. degree in System and Computer Engineering from Carleton University in 2008. She was a network engineer in China Unicom from 2000 to 2005, and a Graduate Research Assistant at Carleton University from 2006 to 2008. She has been working in industry on wireless and communication networks since 2008. Her research interests include wireless sensor networks and RFID.