J Supercomput (2014) 68:87–105
DOI 10.1007/s11227-013-1024-6

Approximation schemes for load balanced clustering in wireless sensor networks

Pratyay Kuila · Prasanta K. Jana

Published online: 28 September 2013

Abstract Clustering sensor nodes is an efficient technique to improve scalability and life time of a wireless sensor network (WSN). However, in a cluster based WSN, the leaders (cluster heads) consume more energy due to some extra load for various activities such as data collection, data aggregation, and communication of the aggregated data to the base station. Therefore, balancing the load of the cluster heads is a crucial issue for the long run operation of the WSNs. In this paper, we first present a load balanced clustering scheme for wireless sensor networks. We show that the algorithm runs in $O(n \log n)$ time for n sensor nodes. We prove that the algorithm is optimal for the case in which the sensor nodes have equal load. We also show that it is a polynomial time 2-approximation algorithm for the general case, i.e., when the sensor nodes have variable load. We finally improve this algorithm and propose a 1.5-approximation algorithm for the general case. The experimental results show the efficiency of the proposed algorithm in terms of the load balancing of the cluster heads, execution time, and the network life.

Keywords Clustering · Load balancing · Approximation algorithm · Network lifetime

1 Introduction

The wireless sensor networks (WSNs) have been paid enormous attention for their potential use in monitoring environment, disaster management, combat field reconnaissance, military surveillance, health, home applications, etc. [1, 2]. A WSN is

P. Kuila · P.K. Jana (✉)
Department of Computer Science and Engineering, Indian School of Mines, Dhanbad 826 004, India
e-mail: prasantajana@yahoo.com

P. Kuila
e-mail: pratyay_kuila@yahoo.com

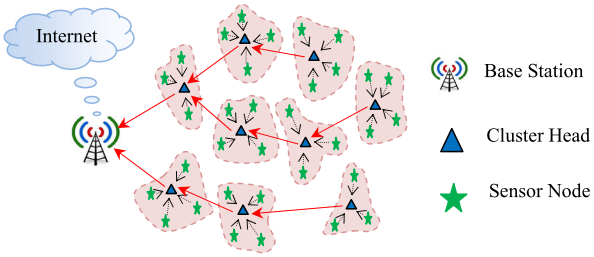

Fig. 1 A model of cluster based WSN. An *arrow* from a sensor node towards a cluster head indicates that the sensor node has been assigned to the cluster head

composed of a large number of sensor nodes, which are randomly or manually deployed in some target area. The sensor nodes collect local information, process them, and send it to a remote base station called sink. The sensor nodes are low power battery operated and their replacement is almost impossible in a harsh environment. Therefore, energy consumption of the sensor nodes is the most important issue in the design of a WSN. Reducing energy consumption is thus considered as the most critical challenge in order to maximize the network lifetime. Many research schemes [3, 4] have been addressed on this issue. However, efficient clustering algorithm for WSN is the most focused area, which has drawn much attention in the research community. In a cluster based WSN, sensor nodes are grouped into distinct clusters. Every cluster has a leader, called cluster head (CH). Each sensor node belongs to only one cluster. The sensor nodes inside a cluster collect data and communicate with their CH. The CHs collect and process the local data and send them to the sink by single hop or multihop communication through other CHs. An example of a cluster based WSN is shown in Fig. 1.

A cluster based WSN has many advantages as follows [5]:

(1) It can reduce energy consumption significantly as only one representative (i.e., CH) per cluster needs to be involved in data aggregation and routing process.
(2) It can considerably conserve communication bandwidth as the sensor nodes need to communicate with their CHs only, and thus can avoid exchange of redundant messages.
(3) The clusters can be more easily managed as they can localize the route set up and require small routing tables for the sensor nodes. This in turn improves the scalability of the network significantly.

However, in a cluster based WSN, CHs bear some extra work load which is contributed by their member sensor nodes as follows. (1) CHs communicate with all the sensor nodes within their cluster, (2) they perform data fusion to discard redundant and uncorrelated data sent by their member sensor nodes, and finally (3) they send the processed data to the sink. Moreover, in many WSNs the CHs are usually selected from the normal sensor nodes, which can die quickly owing to this extra work load. In this context, many researchers [6–10] have proposed the use of some special nodes called gateways or relay nodes, which are provisioned with extra energy. These gateways are treated as the cluster heads and, therefore, can be used interchangeably for the rest of the paper.

It is important to note that the gateways are also battery operated, and hence power constrained. Life time of the gateways is very crucial for long run operation of the

network. Therefore, improper cluster formation may cause some CHs (gateways) overloaded. Such overload may increase latency in communication, consumes high energy of the CH and degrade the overall performance of the WSN. Therefore, load balancing of the CHs is the most important issue for clustering sensor nodes. Particularly, this is a pressing issue when the sensor nodes having unequal load are not distributed uniformly. It is also noteworthy that for a WSN with n sensor nodes and m gateways, the number of possible clusters is m^n. This implies that the computational complexity of finding the optimal load balanced clustering for a large WSN seems to be very high by a brute force approach. In fact, load balanced clustering with unequal load of the sensor nodes is a NP-hard problem.

In this paper, we are concerned with assigning the sensor nodes to the CHs to form clusters such that the maximum load of each CH is minimized. By the load of the CHs, we mean that the load contributed by the member sensor nodes due to data generation and communication. In order to search faster and improved algorithms, we first propose here a load balanced clustering algorithm that produces optimal solution in $O(n \log n)$ time for a special case in which all sensor nodes have equal load. We then show that the same algorithm can work for the general case, i.e., sensor nodes having unequal loads. We prove that this is actually a 2-approximation algorithm. Finally, we present an improved algorithm for the general case which is 1.5-approximation. This algorithm is also shown to run in $O(n \log n)$ time. The experimental results show that the proposed algorithms are better than the existing algorithms in terms of load balancing, execution time, and the network life in rounds. Therefore, our contributions in this paper can be summarized as follows:

- A load balanced clustering algorithm for both equal and unequal load of the sensor nodes in $O(n \log n)$ time for n number of sensor nodes.
- Proof of the algorithm for optimal solution for equal load of the sensor nodes and proof of the 2-approximation algorithm for unequal load.
- A 1.5-approximation load balanced clustering algorithm having same time complexity and its proof.
- Simulation results to demonstrate that the proposed algorithms are superior to existing algorithms.

The rest of the paper is organized as follows. The related work is presented in Sect. 2. System model and problem formulation are described in Sect. 3. Section 4 presents the used terminologies of the algorithms. The proposed algorithm for equal loads and the 1.5-approximation algorithm for unequal loads are presented in Sect. 5 and Sect. 6, respectively. Experimental results are given in Sect. 7 and we conclude our paper in Sect. 8.

2 Related work

A number of clustering algorithms have been developed for WSN [5, 11–13]. LEACH [14] is a popular technique that forms clusters by using a distributed algorithm. It dynamically rotates the work load of the CH among the sensor nodes, which is useful for load balancing. However, the main disadvantage of this approach is that a node with

very low energy may be selected as a CH which may die quickly. Moreover, the CHs communicate with base station via single-hop, which is impractical for WSNs with large coverage area. Therefore, a large number of algorithms have been developed to improve LEACH such as PEGASIS [15], AEEC [16], etc. Compared to LEACH, PEGASIS improves network lifetime, but it requires dynamic topology adjustment and the data delay is significantly high which is unsuitable for large scale networks. Buyanjargal et al. [16] have presented AEEC, a modified algorithm of LEACH. Although AEEC performs clustering phase similar to LEACH, it selects the CHs based on the remaining energy of the sensor nodes and thus overcomes the problem of random selection of CHs. However, the main drawback of this protocol is a high volume of control massage exchange between elected CH and the sensor nodes to form a cluster. Bandyopadhyay et al. [17] have developed a multihop hierarchical clustering algorithm. Their method is based on probabilistic approach that provides optimal value for CH selection and maximum number of hops from a sensor node to cluster head. But their approach does not take into account the residual energy of the sensor nodes and the CH selection may result in faster death of some sensor nodes. Xue et al. [18] have proposed a clustering algorithm by determining the optimal cluster size and presented a location aware hybrid transmission scheme. But it is difficult to control the actual size of the clusters when the number of dead nodes is increased. It is also cumbersome to maximize the network lifetime through control distribution of the cluster heads. Recently, we have proposed a GA based load balanced clustering algorithm for WSNs in [19]. The algorithm forms clusters in such way that the maximum load of each gateway is minimized. As it is a meta-heuristic approach, it may not provide an optimal solution for the equal load of the sensor nodes. We have also developed an energy efficient load-balanced clustering algorithm (EELBCA) [6] that runs in $O(n \log m)$ time. EELBCA addresses energy efficiency as well as load balancing. EELBCA is a min-heap based clustering algorithm. A min-heap is build using cluster heads (CHs) on the number of sensor nodes allotted to the CHs. However, it does not consider unequal load of the sensor nodes. Gupta et al. [7] proposed a load balanced clustering algorithm called LBC, where the authors define cardinality of a cluster as the number of sensor nodes associated with the cluster and attempts to minimize the variance of the cardinality of each cluster in the system. LBC takes $O(mn \log n)$ time in worst case for n number of sensor nodes and m number of CHs in the networks. Tarachand et al. [20] presented a centralized energy efficient load balancing algorithm called CELBA, which takes care of both load balancing and energy consumption of the sensor nodes. The algorithms presented in [7] and [20] assume that all sensor nodes generates equal traffic load to its CHs. Unfortunately, the algorithms may not be effective for the network that generates unequal traffic load to the CHs. Low et al. [8] proposed two different clustering algorithms. Their first algorithm called LBCA (load balanced clustering algorithm) assumes a special case in which the traffic load contributed by all sensor nodes is equal. To form cluster, LBCA constructs an individual BFS (Breadth First Search) tree for each sensor node to find out the least loaded CH for assigning the sensor node to the least loaded CH. However, the time complexity of LBCA is $O(mn^2)$, which may not be effective for large scale WSNs. Moreover, for a large WSN, building, a BFS tree for individual sensor node takes a substantial amount of memory space. They also proposed an approximation algorithm called GLBCA (greedy load balanced-clustering algorithm)

for the case where sensor nodes have unequal loads and this algorithm is shown to be 1.5-approximation. To form a cluster, GLBCA considers a bipartite graph of the sensor nodes and the CHs to find out the maximum matching for assigning a sensor node to a CH. The time complexity of the algorithm is $O(n[n+m+q])$ where q is the cardinality of the bipartite graph. The algorithms proposed in this paper have the following improvements over the Low's algorithm [8].

- The algorithms work for both equal and unequal load of the sensor nodes.
- Both the algorithms run in $O(n \log n)$ time in contrast to $O(n[n+m+q])$ and $O(mn^2)$ time for unequal and equal loads, respectively [8].
- The algorithm requires maintaining only liner array of sensor nodes in contrast to a BFS tree and bipartite graph as required by [8]. Thus, it has improved space complexity.
- The proposed 1.5 approximation algorithm for unequal load shows better load balancing and execution time.

3 System model and problem formulation

We assume a WSN model where all the sensor nodes are randomly deployed along with a few gateways and once they are deployed, they become stationary. A sensor node can be assigned to any gateway if it is within the communication range of the sensor node. Therefore, there are some prespecified gateways onto which a particular sensor node can be assigned. Thus each sensor node has a list of gateways and it can be assigned to only one gateway amongst them. Similar to LEACH, the data gathering operation is divided into rounds. In each round, all sensor nodes sense local data and send it to their CH. Then CHs perform data aggregation to discard the redundant and uncorrelated data and send the aggregated data to the base station. Between two adjacent rounds, all nodes turn off their radios to save energy. All communication is over wireless link. A wireless link is established between two nodes only if they are within the communication range of each other. Current implementation supports TDMA [21–23] protocol to provide MAC layer communication. Gateways use slotted carrier-sense multiple access (CSMA) MAC [23] protocol to communicate with base station. We use the following notations for the problem formulation as follows:

- Let $S = \{s_1, s_2, \ldots, s_n\}$ be the set of sensor nodes and $\zeta = \{g_1, g_2, \ldots, g_m\}$ be the set of gateways where, $n > m$.
- $d_{\max}$ denotes the maximum communication range of the sensor nodes and $dist(i, j)$ denotes the Euclidian distance between sensor node s_i and gateway g_j.
- d_i denotes the traffic load contributed by a sensor node s_i, $s_i \in S$, $d_i \in \mathbb{Q}$ where $\mathbb{Q}$ is the set of rational numbers. This may be noted that the sensor nodes may have practically different processing and communication capabilities for WSNs, which are heterogeneous in nature. Therefore, the traffic load contributed by the sensor nodes may vary depending on rate of data generation and communication. We assume that the traffic load contributed by each sensor node is estimated prior to the formation of clusters as assumed in [8].

- G_i denotes the set of gateways, which are within communication range of node s_i. Therefore, s_i can be assigned to any one of the gateway from G_i, where $G_i \subseteq \zeta$. For example, $G_r = \{g_1, g_3, g_7\}$ means that s_r can be assigned to any one of the CHs, g_1, g_3, g_7.
- Let W_j be the load assigned to the cluster head g_j. In other words,

$$W_j = \sum_{i=1}^{n} d_i \times \alpha_{ij} \tag{3.1}$$

where α_{ij} is a Boolean variable such that

$$\alpha_{ij} = \begin{cases} 1, & \text{if sensor node } s_i \text{ is assigned to cluster head } g_j \\ 0, & \text{otherwise} \end{cases}$$

Then the overall maximum load of cluster heads is $W = \max\{W_i | \forall g_i \in \zeta\}$. Now, we address the problem of clustering, where our main objective is to minimize the overall maximum load of the CHs. Then the Integer Linear Programming (*ILP*) of the load-balanced clustering problem can be formulized as follows:

$$\text{Minimize} \quad W = \max\{W_i | \forall g_i \in \zeta\}$$

Subject to

$$\sum_{j=1}^{m} a_{ij} = 1 | \forall s_i \in S \tag{3.2}$$

$$\sum_{s_i \in S} d_i \times a_{ij} \leq W | \forall g_j \in G_i \tag{3.3}$$

$$\sum_{j=1}^{m} dist(i, j) \times a_{ij} \leq d_{\max} | \forall s_i \in S \tag{3.4}$$

The constraint (3.2) states that a sensor node can be assigned to one and only one gateway and (3.3) indicates that the total load of all the sensor nodes assigned to a gateway must not exceed the overall maximum load of the gateway. The constraint (3.4) ensures that the sensor nodes are assigned to the gateway within their communication range.

4 Terminologies

We use the following terminologies in the proposed algorithms. Depending on the communication range between the sensor nodes and the gateways, we define two kinds of sensor nodes in the system: the *restricted node* and *open node* as follows [6].

(1) *Restricted Node and Restricted Set*: Restricted nodes are those sensor nodes, which can communicate with one and only one gateway. Restricted set is the

set of all restricted nodes in the WSN. We refer this set as "R_{set}." It is obvious to note that a sensor node s_i belongs to R_{set}, if it satisfies the following criteria:

$$s_i \in R_{\text{set}} \Leftrightarrow \left[\left\{\exists g_j | g_j \in G_i \wedge g_j \in \zeta\right\} \wedge \left\{\exists(\neg g_k) | g_k \in G_i \wedge g_k \in (\zeta - g_j)\right\}\right]$$

where G_i is the set of all those gateways, which are within communication range of s_i and ζ is the set of all gateways as mentioned above.

(2) *Open Node and Open Set*: Open nodes are those sensor nodes, which can communicate with more than one gateway. Open set is the collection of all open nodes in the WSN. We refer this set as "O_{set}." A sensor node s_i belongs to O_{set}, if it satisfies the following criteria:

$$s_i \in O_{\text{set}} \Leftrightarrow [s_i \notin R_{\text{set}}]$$

(3) Assigning a sensor node to a CH depends on various factors such as its distance from the CH, its rate of data generation and communication to the CH and its residual energy. We use $P_i(g_x)$ to denote the probability of assigning s_i to the cluster head g_x. Therefore, we have

$$\sum_{x=1}^{m} P_i(g_x) = 1 | \forall s_i \in S, \quad g_x \in \zeta \tag{4.1}$$

(4) The expected load (*EL*) of a cluster head g_x is defined as the summation of mean loads of all the sensor nodes with the probability that they can be assigned to g_x. Therefore, the *EL* can be expressed as follows:

$$EL(g_x) = \sum_{i=1}^{n} P_i(g_x) \times d_i | \forall s_i \in S, \quad g_x \in G_i \tag{4.2}$$

We illustrate it with the following example. Let us assume that there are three sensor nodes s_1, s_2 and s_3 and their set of possible CHs are $G_1 = \{g_1, g_4, g_5\}$, $G_2 = \{g_2, g_4, g_7, g_{10}\}$ and $G_3 = \{g_2, g_4\}$. Here, each of G_1, G_2, and G_3 has g_4 as a common member. Let g_4 is not a member of any other set of CHs except G_1, G_2, and G_3. Therefore, $P_1(g_4) = 1/3$. Similarly, $P_2(g_4) = 1/4$ and $P_3(g_4) = 1/2$. Then using Eq. (4.2), we have

$$EL(g_4) = \left(\frac{1}{3}\right) \times d_1 + \left(\frac{1}{4}\right) \times d_2 + \left(\frac{1}{2}\right) \times d_3$$

If any sensor node, say s_i is finally assigned to the CH, say g_x. Then the *EL* of the cluster head g_x and the *EL* of any other CH, say g_y can be updated as follows:

$$EL(g_x) = EL(g_x) + \left(1 - P_i(x)\right) \times d_i \tag{4.3}$$

$$EL(g_y) = EL(g_y) - P_i(y) \times d_i, \quad \forall g_y \in G_i - \{g_x\} \tag{4.4}$$

(5) Maximum Possible Load (*MPL*) of a cluster head g_x is the summation of the loads contributed by all the sensor nodes prior their allotment to g_x. This can be

noted that g_x must be within the range of all such sensor nodes. Therefore, the *MPL* of a cluster head g_x can be expressed as follows:

$$MPL(g_x) = \sum_{i=1}^{n} d_i | \forall s_i \in S \wedge g_x \in G_i \tag{4.5}$$

(6) The Current Load (*CL*) of a cluster head g_x is the summation of the loads contributed by all sensor nodes after their allotment to g_x.

5 Proposed load balanced clustering algorithms

Network setup is performed in two phases: bootstrapping and clustering. During the bootstrapping process, all the sensor nodes and gateways are assigned unique IDs. Then the sensor nodes broadcast their IDs using CSMA/CA MAC layer protocol. Therefore, the gateways within the communication range of these sensor nodes can collect the sensor IDs, and finally send the local network information to the base station. Thus, for each sensor node, the number of gateways within its communication range can be calculated by the base station. In clustering phase base station executes the clustering algorithm. When the clustering is over, all the sensor nodes are informed about the ID of the gateway they belong to.

5.1 Algorithm for equal load

We consider here that each sensor node has equal traffic load, i.e., $d_i = \alpha$ (say), $\forall s_i \in S$, $1 \leq i \leq n$. Therefore, minimizing the overall maximum load of each CH is equivalent to minimizing the maximum number of sensor nodes that can be assigned to each CH. The basic idea of our algorithm is as follows. We first sort the set of sensor nodes S in nondecreasing order on $|G_i|$ of s_i, $\forall s_i \in S$, $1 \leq i \leq n$. Let, $S = \{s_a, s_b, s_c, \ldots, s_p\}$ be this sorted sensor list. We now successively consider each sensor node from this list (starting with s_a) to assign it to the correct CH. In order to assign s_a, we consult its corresponding set of possible CHs, i.e., G_a and calculate the *EL* values using Eq. (4.2) for all the CHs belongs to G_a. The sensor node s_a is assigned to that CH, which has the minimum *EL* value. In other words, s_a selects the cluster head say g_x only if

$$EL(g_x) = \min\{EL(g_k) | \forall g_k \in G_a\} \tag{5.1}$$

If there are two or more CHs with the same *EL* value then, select that CH for which the probability of assigning the sensor node is maximum. If the probability also ties, then select the CH that has minimum number of sensor nodes already assigned to it. After each assignment of sensor, the *EL* value of the CHs is updated by Eqs. (4.3) and (4.4) for the assignment of the next sensor from the sorted list. The same procedure is continued until all the sensor nodes are allotted to their correct CH. The algorithm is formally presented in Fig. 2.

Algorithm: 2-Approx-Load-Balanced-Clustering (2ALBC)

Input: (1) A set of sensor nodes $S = \{s_1, s_2 ..., s_n\}$ with traffic load $d_i, \forall i, 1 \le i \le n$ and the set of cluster heads $\zeta = \{g_1, g_2 ..., g_m\}$. (2) For each s_i, the set of cluster heads G_i and the probability $P_i(x)[\equiv P_i(g_x)], \forall g_x \in G_i$.

Output: An assignment $A : S \to \zeta$ such that the overall maximum traffic loads of CHs is minimized.

==========

Step 1: Sort the sensor nodes in non-decreasing order on the number of possible CHs upon which they can be assigned.

Step 2: ***for*** j = 1 to n ***do***

2.1: $EL[k] = EL[k] + P_j(k) \times d_j, \ \forall g_k \in G_j$ /* $EL[k] \equiv EL[g_k]$ */

end

Step 3: ***for*** j = 1 to n ***do*** /* Assignment of sensor nodes to CHs */

3.1: Select the CH (say g_k) with minimum EL value using equation (4.2) and assign s_j to g_k. If there are more than two CHs with same EL value, then select that CH with highest probability to join. If the probability is also same then select the CH that has minimum number of sensor nodes already been assigned to it.

3.2: $EL[k] = EL[k] + (1 - P_j(k)) \times d_j$

3.3: $EL[x] = EL[x] - P_j(x) \times d_j, \forall g_x \in \{G_j - g_k\}$

end

Step 4: Stop.

Fig. 2 2-Approximation load-balanced clustering algorithm

5.2 An illustration

Consider a WSN of 15 sensor nodes and 4 gateways, i.e., $S = \{s_1, s_2, \ldots, s_{15}\}$ and $\zeta = \{g_1, g_2, g_3, g_4\}$. Without loss of generality, let us assume that $d_i = 1$. Table 1 shows the possible gateways to which the sensor nodes may be assigned. In our algorithm, sensor nodes are sorted in nondecreasing order on G_i. The sorted list of the sensor nodes is shown in Table 2.

Initially, every CH has no load. As seen from the Table 2, s_2 is the first sensor node, which can be assigned to any one of the cluster heads g_1 and g_2. Therefore, we calculate the EL values of g_1 and g_2 as follows. We observe from Table 2 that the possible sensor nodes that may be assigned to g_1 are s_1, s_2, s_4, s_6, s_7, s_8, s_9, s_{11}, s_{12}, $s_{14,}$ and s_{15}. It is also clear from Table 2 that s_1 has four possible CHs. So, the probability of assigning s_1 to g_1 is 1/4, i.e., $P_1(g_1) = 1/4$. Similarly, the probability of assigning s_2, s_4, s_6, s_7, s_8, s_9, s_{11}, s_{12}, $s_{14,}$ and s_{15} to g_1 are 1/2, 1/4, 1/4, 1/3, 1/3, 1/2, 1/4, 1/4, 1/2, and 1/3, respectively. Therefore, by Eq. (4.2), we obtain

$$EL(g_1) = \left(\frac{1}{4}.1 + \frac{1}{2}.1 + \frac{1}{4}.1 + \frac{1}{4}.1 + \frac{1}{3}.1 + \frac{1}{3}.1 + \frac{1}{2}.1 + \frac{1}{4}.1 + \frac{1}{4}.1 + \frac{1}{2}.1 + \frac{1}{3}.1\right)$$

$$[\because d_i = 1] = 3.75$$

Table 1 Sensor nodes with the list of possible gateways

Sensor nodes (s_i)	Possible gateways (G_i)
s_1	$G_1 = \{g_1, g_2, g_3, g_4\}$
s_2	$G_2 = \{g_1, g_2\}$
s_3	$G_3 = \{g_2, g_3\}$
s_4	$G_4 = \{g_1, g_2, g_3, g_4\}$
s_5	$G_5 = \{g_2, g_3, g_4\}$
s_6	$G_6 = \{g_1, g_2, g_3, g_4\}$
s_7	$G_7 = \{g_1, g_2, g_3\}$
s_8	$G_8 = \{g_1, g_2, g_4\}$
s_9	$G_9 = \{g_1, g_3\}$
s_{10}	$G_{10} = \{g_2, g_3, g_4\}$
s_{11}	$G_{11} = \{g_1, g_2, g_3, g_4\}$
s_{12}	$G_{12} = \{g_1, g_2, g_3, g_4\}$
s_{13}	$G_{13} = \{g_2, g_3\}$
s_{14}	$G_{14} = \{g_1, g_2\}$
s_{15}	$G_{15} = \{g_1, g_3, g_4\}$

Table 2 Sorted list of sensor nodes with possible gateways

Sensor nodes (s_i)	Possible gateways (G_i)
s_2	$G_2 = \{g_1, g_2\}$
s_3	$G_3 = \{g_2, g_3\}$
s_9	$G_9 = \{g_1, g_3\}$
s_{13}	$G_{13} = \{g_2, g_3\}$
s_{14}	$G_{14} = \{g_1, g_2\}$
s_5	$G_5 = \{g_2, g_3, g_4\}$
s_7	$G_7 = \{g_1, g_2, g_3\}$
s_8	$G_8 = \{g_1, g_2, g_4\}$
s_{10}	$G_{10} = \{g_2, g_3, g_4\}$
s_{15}	$G_{15} = \{g_1, g_3, g_4\}$
s_1	$G_1 = \{g_1, g_2, g_3, g_4\}$
s_4	$G_4 = \{g_1, g_2, g_3, g_4\}$
s_6	$G_6 = \{g_1, g_2, g_3, g_4\}$
s_{11}	$G_{11} = \{g_1, g_2, g_3, g_4\}$
s_{12}	$G_{12} = \{g_1, g_2, g_3, g_4\}$

Similarly, we calculate $EL(g_2) = 4.583$. As g_1 has lesser EL value than g_2, we assign s_2 to g_1. After assigning, $EL(g_1)$ is updated by Eq. (3.3) as follows:

$$\begin{aligned} EL(g_1) &= EL(g_1) + \big(1 - P_2(g_1)\big).1 \\ &= 3.75 + 0.5 \quad \big[\because P_2(g_1) = 1/2\big] \\ &= 4.25 \end{aligned}$$

Table 3 Assignment of successive sensor nodes to the CHs

Sensors with possible cluster heads	Cluster head (g_i)			
	g_1	g_2	g_3	g_4
	Assigned sensors and *EL*	Assigned sensors and *EL*	Assigned sensors and *EL*	Assigned sensors and *EL*
$s_2\{g_1, g_2\}$	s_2, $EL = 4.25$	$EL = 4.083$	$EL = 4.083$	$EL = 2.583$
$s_3\{g_2, g_3\}$	s_2, $EL = 4.25$	s_3, $EL = 4.583$	$EL = 3.583$	$EL = 2.583$
$s_9\{g_1, g_3\}$	s_2, $EL = 3.75$	s_3, $EL = 4.583$	s_9, $EL = 4.083$	$EL = 2.583$
$s_{13}\{g_2, g_3\}$	s_2, $EL = 3.75$	s_3, $EL = 4.083$	$s_{13,}s_9$, $EL = 4.583$	$EL = 2.583$
$s_{14}\{g_1, g_2\}$	$s_{2,}s_{14}$, $EL = 4.25$	s_3, $EL = 3.583$	$s_{13,}s_9$, $EL = 4.583$	$EL = 2.583$
⋮	⋮	⋮	⋮	⋮
$s_{12}\{g_1, g_2, g_3, g_4\}$	s_{14}, s_2, s_{15}, s_6, $EL = 4.0$	s_3, s_7, s_{10}, s_{11}, $EL = 4.0$	$s_{13,}s_9, s_1, s_{12}$, $EL = 4.0$	$s_{5,}s_8, s_4$, $EL = 3.0$

The *EL* value of g_2 is also updated by Eq. (4.4) as follows:

$$\begin{aligned} EL(g_2) &= EL(g_2) - P_2(g_2).1 \\ &= 4.583 - 0.5 \quad \left[\because P_2(g_2) = 1/2\right] \\ &= 4.083 \end{aligned}$$

Next, we consider s_3 for its assignment. From Table 2, it can be noted that s_3 may be assigned to g_2 or g_3. Now we calculate *EL* value of g_3 as follows:

$$\begin{aligned} EL(g_3) &= \left(\frac{1}{4}.1 + \frac{1}{2}.1 + \frac{1}{4}.1 + \frac{1}{3}.1 + \frac{1}{4}.1 + \frac{1}{3}.1 + \frac{1}{2}.1 + \frac{1}{3}.1 + \frac{1}{4}.1 + \frac{1}{4}.1 + \frac{1}{2}.1\right) \\ &= 4.083 \end{aligned}$$

As the *EL* values of g_2 and g_3 are same, $P_3(g_2)$ and $P_3(g_3)$ are also same and both g_2 and g_3 are zero loaded, s_3 can be assigned to any one of g_2 and g_3. We assign it to g_2. Now we update *EL* values of g_2 and g_3 and continue the same method for assigning the remaining sensor nodes. The successive assignment of the sensor nodes s_3, $s_{9,}$ and finally for s_{12} to their CHs along with their calculated *EL* values are shown in Table 3.

Lemma 5.1 *The above* 2*-approx-load-balanced-clustering algorithm* (2*ALBC*) *produces optimal solution for equal load of the sensor nodes.*

Proof First we note that the sensor list S is initially sorted in non-decreasing order on the number of CHs. Let $S = \{s_1, s_2, s_3, \ldots, s_n\}$ be this sorted list. Now, the algorithm assigns $s_1, s_2, s_3, s_4, \ldots, s_n$ successively. For the assignment, the following approach is followed for load balancing. The algorithm assigns that sensor node first, which has the least chance of assigning to a CH. As a result any other sensor node having more chance can select a CH with the least load.

Let s_k be the last assigned sensor node to the maximum loaded cluster head g_r after its assignment. Let current load on g_r be l_r. Suppose there is a cluster head $g_s \in G_k$ with total load l_s such that $l_s < l_r$. Therefore, s_k can be assigned to g_s and current load of g_s will be $l_s + \alpha$, where α is the load of each sensor node. Then the algorithm is not optimal if $(l_s + \alpha) < l_r$. But at the time of assignment of s_k, the selected cluster head g_r had the minimum EL value, i.e., $(l_r - \alpha)$ was minimum. So, after assigning of s_k to any $g_s \in G_k - \{g_r\}$, $(l_s + \alpha) \geq l_r$. Hence, the following proof. □

Lemma 5.2 *The time complexity of the algorithm* 2*ALBC is* $O(n \log n)$.

Proof Step 1 requires $O(n \log n)$ time for sorting n sensor nodes. In for loop of the step 2, a CH list is created where the average possible load of each CH is calculated. In the worst case, step 2.1 can iterate m times. So, step 2 can take $O(mn)$ time in worst case. Step 3 iterates n times in which step 3.1 and step 3.3 are the dominating ones requiring $O(m)$ time for each in the worst situation. Therefore, step 3 can be executed in $O(mn)$ time. Thus, the above algorithm requires $O(mn)$ time. If $m < \log n$, it requires $O(n \log n)$ time. □

Theorem 1 *The algorithm* 2*ALBC produces optimal solution in* $O(n \log n)$ *time assuming equal loads of the sensor nodes.*

Proof Follows from Lemmas 5.1 and 5.2. □

5.3 Approximation algorithm for unequal load

We consider here the following scenario of the WSN in which the traffic load contributed by the sensor nodes are variable. This is due to the fact that the sensor nodes may generate or process the data at different rate and their rate of communication with their CHs is also different.

Lemma 5.3 *Load balanced clustering problem with unequal load for sensor nodes is NP-hard.*

Proof Please refer [8]. □

Lemma 5.4 *The algorithm* 2*ALBC is a* 2*-approximation algorithm for load balanced clustering problem with unequal load of the sensor nodes.*

Proof Let OPT be the maximum load of a CH in an optimal solution. Then it is obvious to note that $OPT \geq d_i$, $\forall i, 1 \leq i \leq n$. Let I be the smallest instance of the problem for which the proposed algorithm produces results not as good as the optimal solution. Let g_i be a CH with maximum load after complete run of the load-balanced- clustering algorithm, i.e., $W_i = \max\{W_j | \forall j, 1 \leq j \leq n\}$. Let s_r be the last sensor node assigned to g_i. The crucial property of our algorithm is that, at the time

of the assignment of s_r, g_i was the minimum loaded CH from G_r. So, before assignment of s_r, load of g_i was $(W_i - d_r)$, which is less than or equal to $OPT(I)$. Therefore, $W_i - d_r \leq OPT(I)$, i.e., $W_i \leq OPT(I) + OPT(I)$ (as, $d_r \leq OPT(I)$). Hence, $W_i \leq 2OPT(I)$. □

6 Proposed 1.5-approximation algorithm

We assume here that the traffic loads contributed by the sensor nodes are unequal and already calculated prior cluster formation. The basic idea is as follows. We first assign all the sensor nodes $s_i, \forall s_i \in R_{\text{set}}$, to their corresponding gateway. Then all the sensor nodes $s_j, \forall s_j \in O_{\text{set}}$, are sorted in nonincreasing order on their contributing traffic load. Now, the basic principle of the proposed algorithm is to assign a sensor node, which contributes maximum load to a minimum loaded CH within its range. After the assignment, current load and maximum possible load of the CHs are updated. Then the sensor node contributing next maximum load is assigned to the current minimum loaded CH within its range. This process is continued until all the sensors are assigned to the CHs. The algorithm is now formally presented in Fig. 3.

Lemma 6.1 *The time complexity of* 1.5*-approx-load-balanced-clustering* (1.5*ALBC*) *algorithm is* $O(n \log n)$.

Proof Step 1 requires $O(m)$ time in worst case to assign the R_{set} to their corresponding CH. In step 2 a CH list is created where the maximum possible load of each CH is calculated. In the worst case, the step 2 can take $O(mn)$ time. Step 3 requires $O(n \log n)$ time in worst case for sorting sensor nodes of O_{set}. In step 4, a sensor node is assigned to a CH. Step 4.1 through step 4.5 can be run in $O(m)$ time. Outer for loop runs in $O(n)$ time. So step 4 can be executed in $O(mn)$ time. Therefore the above algorithm requires $O(mn)$ time. If $m < \log n$, it requires $O(n \log n)$ time. □

Lemma 6.2 *The* 1.5*ALBC is a* 1.5*-approximation algorithm of load balanced clustering problem* (*LBCP*) *with unequal loads of the sensor nodes*.

Proof Let *OPT* be the maximum load of a CH in an optimal solution. It is obvious to note that $OPT \geq d_i$, $\forall i, 1 \leq i \leq n$. Let I be the smallest instance of the problem for which this algorithm conflicts with optimal solution. Let g_i be a CH with maximum load after complete run of the algorithm approx-load-balanced, i.e., $W_i = \max\{W_j | \forall j, 1 \leq j \leq n\}$.

Let s_r be the last sensor node assigned to g_i. Therefore, s_r is the sensor node which has the minimum load between all sensor nodes. Now at the time of the assignment of s_r, g_i was the minimum loaded CH from G_r. As we are assuming that I is the smallest instance, so the algorithm conflicts with optimal solution only after assignment of s_r. Without loss of generality, we may assume that each CH can be assigned with at least two sensor nodes. Therefore, all the CHs except g_i are assigned by at least two sensor nodes. This implies that $d_r \leq OPT(I)/2$, as s_r is the least loaded sensor node.

Algorithm: 1.5-Approx-Load-Balanced-Clustering (1.5ALBC)

Input: (1) A set of sensor nodes $S = \{s_1, s_2 ..., s_n\}$ with traffic load $d_i, \forall i, 1 \le i \le n$ and the set of cluster heads $\zeta = \{g_1, g_2 ..., g_m\}$. (2) For each s_i, the set of cluster heads G_i.

Output: An assignment $A : S \rightarrow \zeta$ such that the overall maximum traffic loads of CHs is minimized.

==========

Step 1: ***While*** ($R_{set} \neq$ ***NULL***) ***do***

1.1: Assign successive sensor node s_i to their corresponding cluster head (say g_k) and delete s_i from R_{set}.

1.2: $CL[k] = CL[k] + d_i$

end

Step 2: ***for*** $j = 1$ to m ***do***

$$MPL[j] = CL[j] + \sum (d_i \,|\, g_j \in G_i \wedge s_i \in O_{set})$$

end

Step 3: Sort the sensor nodes from O_{set} in non-decreasing order on the number of possible CHs upon which they can be assigned.

Step 4: ***while*** ($O_{set} \neq$ ***NULL***) ***do***

4.1: Select successive sensor node (say s_i) from O_{set}.

4.2: Select the cluster head (say g_k) from G_i with minimum current loaded (CL). If there are more than two CHs with the same current load (CL), then select the CH which has minimum MPL value.

4.3: Assign s_i to g_k and delete s_i from g_k.

4.4: $CL[k] = CL[k] + d_i$

4.5: $MPL[x] = MPL[x] - d_i, \forall g_x \in \{G_i - g_k\}$

Step 5: Stop.

Fig. 3 1.5-Approximation load-balanced clustering algorithm

Therefore, before allotment of s_r, load of g_i was $(W_i - d_r)$, which is less than or equal to $OPT(I)$. In other words, $W_i - d_r \le OPT(I)$, i.e., $W_i \le OPT(I) + OPT(I)/2$ (as, $d_r \le OPT(I)/2$). Hence, $W_i \le 1.5OPT(I)$. □

7 Experimental results

The proposed algorithms were experimented extensively using MATLAB (version 7.5) and C programming language on an Intel Core 2 Duo processor with T9400 chipset, 2.53 GHz CPU and 2 GB RAM running on the platform Microsoft Windows Vista. For the experiments, we assumed two different scenarios. In the first scenario, we considered a 300 × 300 square meter area in which 200 to 500 sensor nodes are randomly deployed along with 35 gateways. We next considered a 500 × 500 square meter area in which 600 to 1200 sensor nodes are randomly deployed along with 75 gateways. For both of the scenarios, base station was assumed to be located at the centre of the area. Each sensor node was assumed to have an initial energy of 2 Joules and gateways with 10 Joules. In the simulation run, the energy model and the typical parameter values were set same as LEACH. For the sake of comparison, we also ran various algorithms including GLBCA [8], GA-based load balanced clustering algorithm [19], LBC [7], and LDC [24].

7.1 Load balancing

To judge the quality of the load balancing, we measured the standard deviation of the gateway load and plotted against the number of sensor nodes. The standard deviation (σ) of the gateway load gives even distribution of the load per gateway. If there are m gateways and n sensor nodes, the standard deviation of gateway load is given by

$$\sigma = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(\mu - W_j)^2}$$

where μ(average load) $= \frac{1}{m}\sum_{i=1}^{n} d_i$, and W_j is the overall load of the gateway g_j.

The results of load balancing for unequal load of the sensor nodes are shown in Figs. 4(a)–4(b). Our proposed algorithms perform better in load balancing issue in this case. However, LBC and LDC perform poorly as they are not proposed for unequal load of the sensor nodes.

We next ran the algorithms for equal load of the sensor nodes, the comparison results of which are depicted in Figs. 5(a)–5(b). This can be noted that although our proposed algorithms 2ALBC and 1.5ALBC perform same as GLBCA, they perform far better than GA, LBC, and LDC. The rationale behind is that all three algorithms 2ALBC, 1.5ALBC, and GLBCA are known to be optimal for the equal load of the sensor nodes whereas the GA, LBC, and LDC are known to be suboptimal in terms of load balancing. Note that LBC can be executed separately for load balancing and energy consumption issues but not considering both of them together. In the simulation, we ran LBC by considering load balancing of the CHs only. LDC performs poorly in load balancing as it does not consider any issue of load balancing of the CHs for cluster formation.

7.2 Execution time

We also obtained the execution time of the algorithms. We can see from Figs. 6(a)–6(b) that the proposed algorithms1.5ALBC and 2ALBC are better than LBC and far better than GLBCA in terms of execution time. The rationale behind this is that in order to assign a sensor node, GLBCA builds a bipartite graph of the sensor nodes and the CHs to find out the maximum matching for assigning a sensor node to a CH. The time complexity of the algorithm is $O(n[n + m + q])$ in worst case (m number of gateways and n number of sensor nodes and q is the cardinality of the bipartite graph). Thus, GLBCA takes considerably high time for large number of sensor nodes. LBC takes time to expands e-Set based on the distance from the sensor node to gateway and in worst case this process takes $O(mn)$ time. Then LBC assigns the e-Sets which also takes $O(mn)$ time and finally assign all remaining sensor nodes which takes $O(mn \log n)$ time. On the other hand, 2ALBC and 1.5ALBC use only sorting and the overall time complexity of these algorithms is $O(n \log n)$ in the worst case. Thus, 2ALBC and 1.5ALBC consume much less execution time than GLBCA and LBC. It can be observed that LDC executes faster than all the algorithms. This is because LDC simply assigns the sensor nodes to its nearest CH, and thus it takes less execution time.

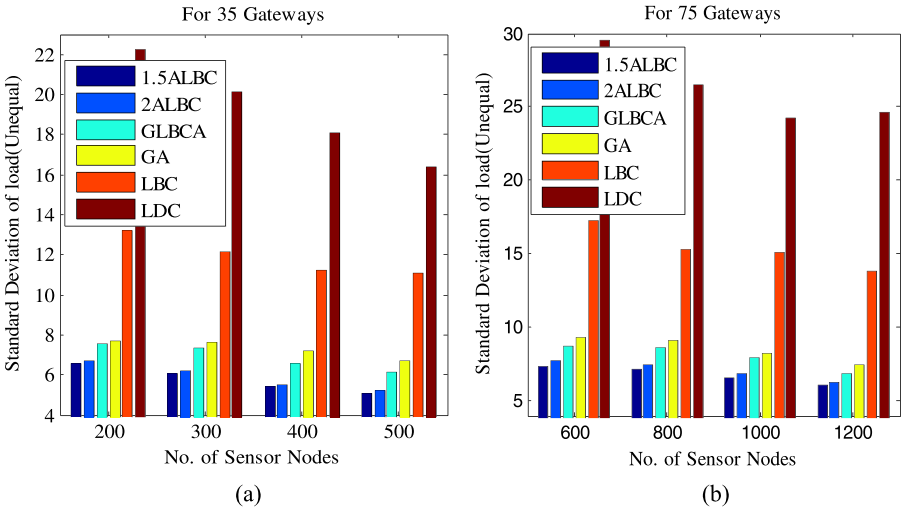

Fig. 4 Comparison of load balancing for unequal load of the sensor nodes using (**a**) 35 gateways and (**b**) 75 gateways

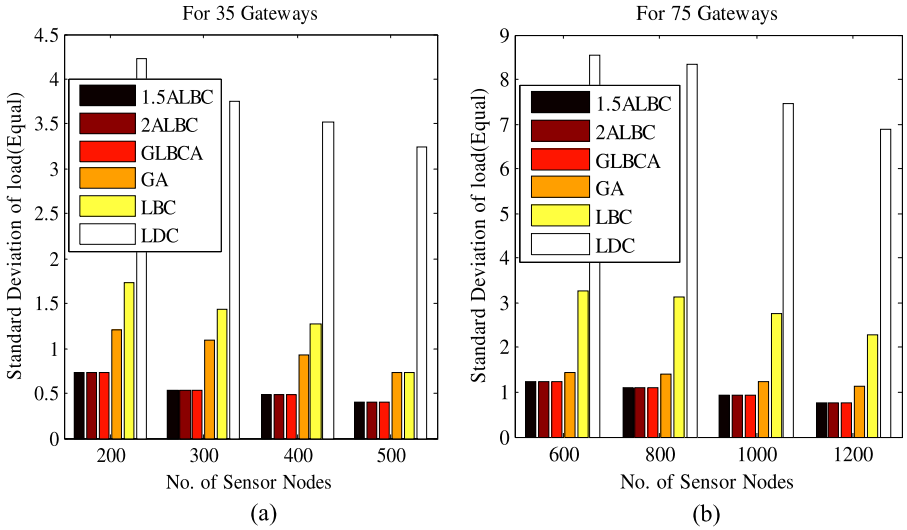

Fig. 5 Comparison of load balancing for equal load of the sensor nodes using (**a**) 15 gateways and (**b**) 30 gateways

7.3 Network life

We now present the experimental results of the algorithms in terms of network life time, which can be defined in various ways as follows. This is the time duration/number of rounds until the first/last node dies or until certain percentage of nodes die [24]. In our experiment, we assumed the network life time as the number

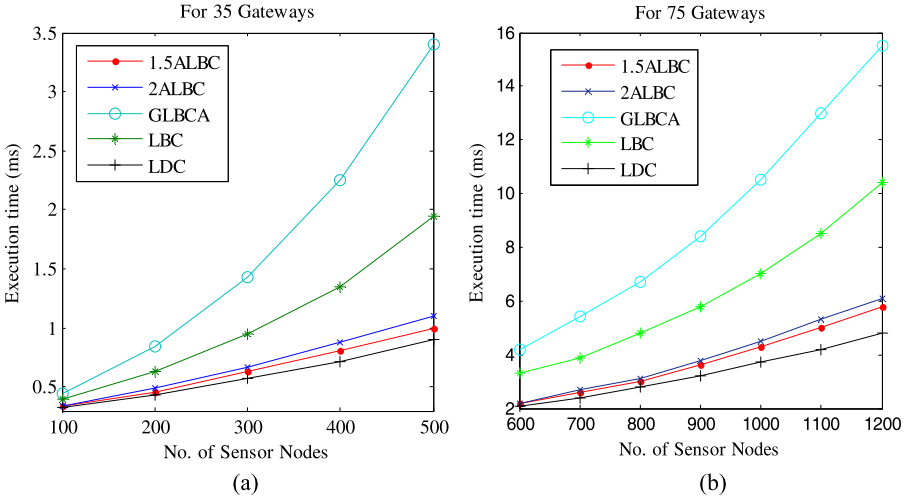

Fig. 6 Comparison of execution time using (**a**) 15 gateways and (**b**) 30 gateways

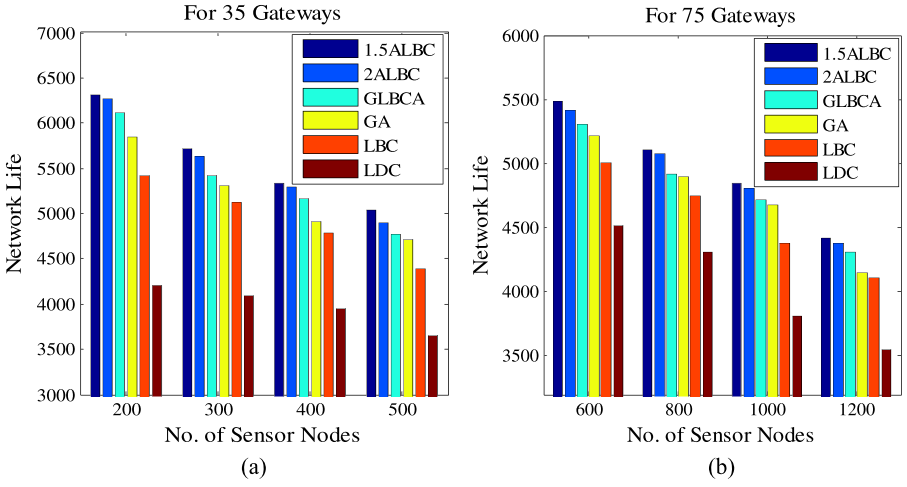

Fig. 7 Comparison of network life time using single hop communication for (**a**) 35 gateways and (**b**) 75 gateways

of rounds until first gateway dies. Therefore, network life can be extended if we can prevent the death of the first gateway. This can be achieved through load balancing. This implies, better the load balancing, higher is the network life. Figures 7(a)–7(b) shows the comparison of the algorithms in terms of the network life time. It is obvious to note that the proposed algorithms perform better. Obviously, LDC performs poorly in this case, also.

In all of the above experiments, we assumed single hop communication between CHs and the base station, which may not be realistic for a large area WSN. In order to test how the proposed algorithms behave for WSN with multihop communication,

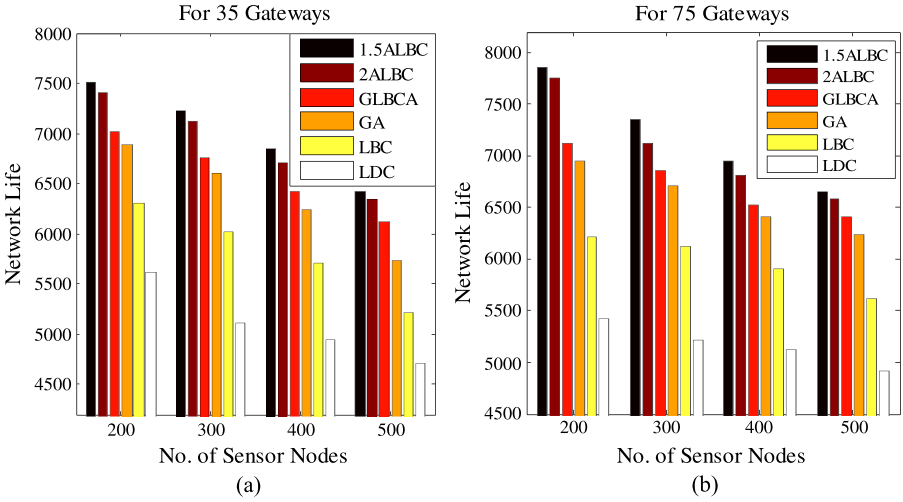

Fig. 8 Comparison of network life time using multi-hop communication for (**a**) 35 gateways and (**b**) 75 gateways

we also performed experiments to measure the network life time. The results are depicted in Fig. 8. We used the same routing scheme for all of the algorithms as proposed in [9]. Note that the proposed algorithms perform better than GLBCA, GA, LBC, and LDC in this scenario, also.

8 Conclusion

In this paper, we have presented two load balanced clustering algorithms for wireless sensor networks. The first algorithm has been shown to be optimal in assigning sensor nodes to the CHs in the case of equal load of the sensor nodes. The algorithm has been shown to run in $O(n \log n)$ time. We have shown that the same scheme can also work as a 2-approximation algorithm for the situation where the loads of the sensor nodes can vary. We have next presented an improved algorithm that has 1.5-approximation ratio for the general case, which is shown to run in $O(n \log n)$ time too. We have shown that the proposed algorithms outperform the existing algorithms in terms of load balancing, execution time and the network life time for both equal and unequal load of the sensor nodes. However, our proposed clustering algorithms consider only the load balancing issue with respect to data generation and communication of the member sensor nodes. We do not consider the residual energy of CHs and the communication distance. Our next attempt will be made to develop a clustering algorithm covering such issues in our future research.

Acknowledgements The first version of the paper was appeared in the proceedings of the international conference ADCONS 2011 (LNCS 7135, pp. 399–404). The authors are thankful to the anonymous reviewers for their valuable suggestions.

References

1. Akyildiz IF et al (2002) Wireless sensor networks: a survey. Comput Netw 38(4):393–422
2. Jennifer Y et al (2008) Wireless sensor network survey. Comput Netw 52(12):2292–2330
3. Giuseppe A et al (2009) Energy conservation in wireless sensor networks: a survey. Ad Hoc Netw 7(3):537–568
4. Emanuele L et al (2007) Energetic sustainability of routing algorithms for energy-harvesting wireless sensor networks. Comput Commun 30:2976–2986
5. Abbasi AA, Younis M (2007) A survey on clustering algorithms for wireless sensor networks. Comput Commun 30:2826–2841
6. Pratyay K, Prasanta KJ (2012) Energy efficient load-balanced clustering algorithm for wireless sensor networks. Proc Technol 6:771–777
7. Gupta G, Younis M (2003) Load-balanced clustering of wireless sensor networks. In: IEEE International conference on communications (ICC), vol 3, pp 1848–1852
8. Low CP et al (2008) Efficient load-balanced clustering algorithms for wireless sensor networks. Comput Commun 31(4):750–759
9. Suneet KG, Pratyay K, Prasanta KJ (2013) GAR: an energy efficient GA-based routing for wireless sensor networks. In: LNCS, vol 7753. Springer, Berlin, pp 267–277
10. Pratyay K, Prasanta KJ (2012) Improved load balanced clustering algorithm for wireless sensor networks. In: LNCS, vol 7135. Springer, Berlin, pp 399–404
11. Olutayo B et al (2010) A survey on clustering algorithms for wireless sensor networks. In: 13th int conf on network-based information systems, pp 358–364
12. Congfeng J et al (2009) Towards clustering algorithms in wireless sensor networks-a survey. In: IEEE wireless communications and networking conference, pp 1–6
13. Pratyay K, Prasanta KJ (2012) An energy balanced distributed clustering and routing algorithm for wireless sensor networks. In: PDGC 2012, pp 220–225
14. Heinzelman WB et al (2002) Application specific protocol architecture for wireless microsensor networks. IEEE Trans Wirel Commun 1(4):660–670
15. Lindsey S, Raghavendra CS (2003) PEGASIS: power efficient gathering in sensor information systems. In: Proc of the IEEE aerospace conference, vol 3, pp 1125–1130
16. Buyanjargal O, Kwon Y (2010) AEEC-adaptive and energy efficient clustering algorithm for content based wireless sensor networks. In: 2nd international conference on computer science and its applications. IEEE Press, New York, pp 1–6
17. Bandyopadhyay S, Coyle EJ (2003) An energy efficient hierarchical clustering algorithm for wireless sensor networks. In: IEEE INFOCOM, USA, vol 3, pp 1713–1723
18. Xue Q, Ganz A (2004) Maximizing sensor network lifetime: analysis and design guides. In: IEEE, vol 2, pp 1144–1150
19. Pratyay K, Suneet KG, Prasanta KJ (2013) A novel evolutionary approach for load balanced clustering problem for wireless sensor networks. Swarm Evol Comput. doi:10.1016/j.swevo.2013.04.002
20. Tarachand A et al (2012) An energy efficient load balancing algorithm for cluster-based wireless sensor networks. In: IEEE INDICON, pp 1250–1254
21. Jianlin M et al (2007) A TDMA scheduling scheme for many-to-one communications in wireless sensor networks. Comput Commun 30(4):863–872
22. Mario OD, Kin KL (2011) TDMA scheduling for event-triggered data aggregation in irregular wireless sensor networks. Comput Commun 34(17):2072–2081
23. Baronti P et al (2007) Wireless sensor networks: a survey on the state of the art and the 802.15.4 and ZigBee standards. Comput Commun 30:1655–1695
24. Ataul B et al (2008) Clustering strategies for improving the lifetime of two-tiered sensor networks. Comput Commun 31(14):3451–3459