Computers and Electrical Engineering 41 (2015) 177–190

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Energy efficient fault tolerant clustering and routing algorithms for wireless sensor networks ☆

Md Azharuddin, Pratyay Kuila, Prasanta K. Jana *,[1]

Department of Computer Science and Engineering, Indian School of Mines, Dhanbad 826 004, India

ARTICLE INFO

Article history:
Available online 27 August 2014

Keywords:
Wireless sensor networks
Clustering
Routing
Gateways
Fault tolerant
Network life

ABSTRACT

Conservation of energy and fault tolerance are two major issues in the deployment of a wireless sensor network (WSN). Design of clustering and routing algorithms for a large scale WSN should incorporate both these issues for the long run operation of the network. In this paper, we propose distributed clustering and routing algorithms jointly referred as DFCR. The algorithm is shown to be energy efficient and fault tolerant. The DFCR uses a distributed run time recovery of the sensor nodes due to sudden failure of the cluster heads (CHs). It takes care of the sensor nodes which have no CH within their communication range. We perform extensive experiments on the proposed algorithm using various network scenarios. The experimental results are compared with the existing algorithms to demonstrate the strength of the algorithm in terms of various performance metrics.

© 2014 Elsevier Ltd. All rights reserved.

1. Introduction

Wireless sensor networks (WSNs) have gained enormous attention for their wide range of applications such as environmental monitoring, military surveillance, health care, and disaster management [1]. One of the major constraints of WSNs is the limited and generally irreplaceable power sources of the sensor nodes. Even, in many applications, it is impractical to replace the sensor nodes as they work under harsh environment. Therefore, reducing energy consumption of the sensor nodes is considered as the most critical challenge for long run operation of WSNs. Extensive researches have been carried out in designing energy saving protocols which include low-power radio communication hardware, energy-aware MAC protocols, etc. However, energy efficient clustering and routing algorithms [2,3] are the most two promising areas that have been studied extensively for WSNs.

In a cluster based WSN (refer Fig. 1), the sensor nodes are organized into distinct groups, called clusters. Each group has a leader, called cluster head (CH) and each sensor node belongs to one and only one cluster. Clustering WSN has following advantages. (1) It enables data aggregation at cluster head to discard the redundant and uncorrelated data, thereby reducing energy consumption of the sensor nodes. (2) Routing can be more easily managed because only CHs need to maintain the local route setup of other CHs and thus requiring small routing information. This in turn improves the scalability of the network significantly. (3) It also conserves communication bandwidth as the sensor nodes communicate with their CHs only and thus avoid the exchange of redundant messages among themselves. However, in clustering approach, a CH bears some extra work load, i.e., receiving sensed data sent by member sensor nodes, data aggregation and data dissemination to the BS.

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. Sabu Thampi.

* Corresponding author.
E-mail addresses: azhar_ism@yahoo.in (M. Azharuddin), pratyay_kuila@yahoo.com (P. Kuila), prasantajana@yahoo.com (P.K. Jana).
[1] IEEE Senior Member.

http://dx.doi.org/10.1016/j.compeleceng.2014.07.019
0045-7906/© 2014 Elsevier Ltd. All rights reserved.

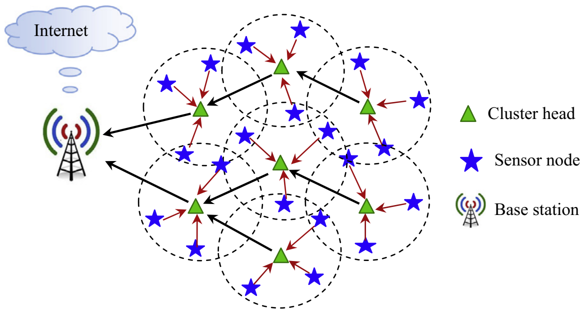

Fig. 1. A model of wireless sensor network.

Moreover, in many WSNs, the CHs are usually selected amongst the normal sensor nodes, which can die quickly as they consume more energy due to such extra work load. In this context, many researchers [4–10] have proposed the use of some special node called gateways or relay nodes that are provisioned with extra energy. These gateways are treated as the cluster heads (CHs) which are responsible for the same functionality of the CHs. Unfortunately the gateways are also battery operated and hence power constrained. Therefore, it is extremely important to properly utilize their energy in the process of both clustering and routing.

Furthermore, sensor nodes are very prone to failure due to several factors such as environmental hazards, energy depletion and device failure. The failure can affect the overall network life time and degrades the overall performance of the network. However, failure of the CHs is catastrophic as it can limit the accessibility of the sensor nodes under their supervision and prevents data aggregation and data dissemination. Therefore, in order to keep WSN operational, the clustering and routing algorithms should cope with the fault tolerant aspects, especially the failure of the CHs.

In this paper, we address the problem of designing energy efficient clustering and routing algorithms for WSNs which are fault tolerant. We present two distributed fault tolerant algorithms, one for clustering and the other for routing in WSNs. The algorithms give emphasis on the failure of the CHs. We name these algorithms as DFCR (Distributed Fault-tolerant Clustering and Routing) in together. The DFCR addresses the issues of energy efficiency. In the clustering phase, sensor nodes select its CH based on a cost function that consists of residual energy of the CH. This is also based on the distance between sensor node to the CH and the distance from the CH to the base station. The algorithm takes care of the sensor nodes which have no CH within their communication range. The DFCR presents a distributed run time recovery of the faulty cluster members due to sudden failure of the CH. To achieve fault tolerance, our method adopts neither redundant deployment of the CHs nor re-clustering approach. In data routing phase, the CHs select their next hop neighbor CH in such a way that their energy consumption will be balanced and minimized. The algorithm can tolerate the sudden failure of CHs in routing path.

We perform extensive experiments on the proposed algorithm through simulation run and compare their results with the fault tolerant clustering algorithm proposed by Gupta and Younis [9], MHRM (Minimum Hop Routing Model) [11] and a distributed routing algorithm DEBR (Distributed Energy Balanced Routing) [12]. Results show that our proposed method performs better than these algorithms with respect to the number of live sensor nodes, number of live gateways, the number of inactive sensor nodes and the energy consumption. Our main contributions are summarized as follows.

- A distributed algorithm for energy efficient fault tolerant clustering for two tire wireless sensor networks.
- A distributed run time recovery of the faulty cluster members due to sudden failure of cluster heads.
- A distributed algorithm for energy efficient fault tolerant routing.
- Experiments of the proposed algorithms through simulation run on different network scenarios.
- Comparison of experimental results to demonstrate superiority of the proposed algorithm over the existing algorithms.

The rest of the paper is organized as follows. The related work is presented in Section 2. The system model is discussed in Section 3 which includes network model, energy model and fault model. The proposed algorithms and the experimental results are presented in Sections 4 and 5 respectively, and Section 6 concludes the paper.

2. Related works

Clustering and routing algorithms have been studied extensively for WSNs. Some of them are centralized approach and the others are distributed. Centralized algorithms are executed by the base station and the clustering or routing information is sent to the gateways (i.e., CHs). However, to execute the algorithm, the base station needs the global information of the network, in contrast to distributed approach which takes decisions based on the local information. Here, we present a review of a few popular centralized and distributed algorithms as follows.

Low et al. [6] have proposed an algorithm which uses a bipartite graph of the sensor nodes and the gateways for finding a maximum matching of assigning a sensor node to a CH. The algorithm has the time complexity of $O(mn^2)$ for n sensor nodes and m CHs. This is very high for a large scale WSN. It also requires building a BFS tree for an individual sensor node which takes a substantial amount of memory space. In [8], we have proposed an algorithm with $O(n \log n)$ time, which is an improvement over [6]. But both the algorithms have not considered the energy consumption issue. In [10], an Energy Efficient Load-Balanced Clustering Algorithm (EELBCA) has been proposed with $O(n \log m)$ time. EELBCA addresses energy efficiency as well as load balancing. EELBCA is a min-heap based clustering algorithm. However, being a centralized approach, re-clustering or routing of the system complicates the network setup. New communication schedule and network setup has to be communicated to the whole network, which may not be realistic for a large area network. The centralized methods are useful where the location of each sensor node and gateway is known to the central controller or base station. Moreover, all of the above algorithms do not address the fault tolerance of the WSNs due to the sudden death of some CHs. A number of distributed algorithms have also been reported for clustering and routing. We describe few popular distributed clustering and routing algorithms as follows.

Low-energy adaptive clustering hierarchy (LEACH) [13] is a popular technique that forms clusters by using a distributed algorithm. It dynamically rotates the work load of the CH amongst 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 the base station via single-hop, which is impractical for a large area network. Therefore, a large number of algorithms have been developed to improve LEACH which can be found in [14,15]. Ok et al. [12] have reported a distributed energy balance routing (DEBR) scheme to balance the energy consumption of the network. However, the main drawback of DEBR is that, for balancing energy consumption it may select a next-hop node which may not have any other next-hop node within its communication range. This stops the dissemination of some precious data to the base station. Moreover, none of the above algorithms have considered fault tolerance issues.

There are some cluster based routing algorithms which consider the fault tolerant issues. Direct diffusion (DD) [16] is a popular multipath routing protocol based on query driven data delivery. Here, multiple node disjoint paths are created between source nodes and the BS. If one of the paths is broken due to intermediate node failure, then another path is chosen to transmit the data. Thus, it increases the reliability in data delivery. However, DD is not suitable for the applications which require continuous data delivery. Moreover, the protocol is not energy efficient as it broadcasts a low rate interest message periodically. Erasure coding [17] is a replication based routing protocol. In this protocol, source node decodes each data packet of size bM bits into M fragments each of size b and generates another K coding fragments to have in total a set of $M + K$ fragments and send it over n multiple paths. But the protocol performs worst when packet loss in the network is very high. Reliable information forwarding (ReInForm) [18] using multipath is another replication based routing protocol. In [5], the authors have presented an ILP formulation to minimize the number of given potential positions for relay nodes placement in the hierarchical network. The relay nodes are placed in such a way that each sensor node is k_s-survivable and each relay node (say R) that is not within transmitting distance of the BS is k_r-survivable. Here, the authors use redundant relay nodes to assure the survivability which leads to bad utilization of the resources. Moreover, the authors assume that relay nodes have some pre assumed potential position to be placed. But in real scenario, nodes are deployed randomly in a harsh environment which cannot even assure the k_s and k_r survivability. Other node deployment algorithms are also developed for WSN which can be seen in [19]. In MHRM (minimum hop routing model) [11], each relay node finds a path to the BS in such a way that the hop count is minimized. Therefore, much amount of energy is spent to transmit data as it selects furthest relay node in its communication range. In [20], authors have developed a distributed fault-tolerant mechanism called CMATO (Cluster-Member-based fAult-TOlerant mechanism) for WSNs. The authors utilize the overhearing techniques to detect the fault. Whenever a CH fails, cluster members detect it and re-select a new CH to form a new cluster. When the link from the CH to the cluster member is broken, then the members transmit the data to the neighboring cluster. Here, the authors assume that clustering is done separately beforehand. Therefore, they do not take care about the cluster formation process. Gupta and Younis [9] have proposed fault tolerant clustering where sensor nodes are recovered in runtime from the faulty cluster. Here, authors assume that each sensor node is within transmission range of at least two CHs. Therefore, whenever a CH fails, the member sensor nodes can join other CH based on communication cost. However, this assumption may not stand in real scenario where sensor nodes are deployed randomly along with the CHs.

Our proposed algorithm has the following advantages over the existing ones.

- It has provided a distributed cluster formation algorithm where the non-CH sensor nodes join a CH depending on their derived cost value. Whereas, in the existing algorithms [13,14] non-CH sensor nodes simply join the CH by considering only received signal strength, which may cause imbalance load of the CHs and may lead to serious energy inefficiency.
- It also takes care of the sensor nodes which have no CH within their communication range in contrast to [5,9,20].
- In order to tolerate the fault, it does not require redundant node deployment [5,19]. Even, it does not require any pre assumed location of the relay nodes to be placed [5]. Therefore, it can work in harsh environment where nodes are deployed randomly.
- In many existing works [5,9,10], the nodes are assumed to be equipped with GPS which is not cost effective in deploying a large scale WSN. Whereas, the proposed algorithm does not require any GPS.

3. System models

3.1. Network model

We assume a WSN model where all the sensor nodes are deployed randomly along with a few gateways and once they are deployed, they become stationary. As a wireless network, the nodes do not have the global information about the network.

However, we consider that the nodes have the local information, i.e., residual energy level and the distance of its neighbors [12,21]. Similar to LEACH [13], the data gathering operation is divided into rounds. In each round, all sensor nodes collect the local data and send it to their corresponding CH (i.e., gateway). On receiving the data, the gateways aggregate them to discard the redundant and uncorrelated data and send the aggregated data to the base station via other CH as a next hop relay node. Between two adjacent rounds, all nodes turn off their radios to save energy. All communication is over wireless links. A wireless link is established between two nodes only if they are within the communication range of each other. Given the transmitting power, a node can compute the distance to another node based on the received signal strength. The same strategy has been adapted in the literature [12,21]. Therefore, it does not require any location finding system such as GPS. The current implementation supports state-of-art MAC protocols [22] to provide MAC layer communication. Gateways use CSMA/CA MAC protocol to communicate with base station [22].

3.2. Energy model

The radio model for energy used in this paper is same as described in [13]. In this model, both the free space and multipath fading channels are used depending on the distance between the transmitter and receiver. When the distance is less than a threshold value d_0, then the free space (*fs*) model is used, otherwise, the multipath (*mp*) model is used. Let E_{elec}, ε_{fs} and ε_{mp} be the energy required by the electronics circuit and by the amplifier in free space and multipath respectively. Then the energy required by the radio to transmit an *l*-bit message over a distance *d* is given as follows.

$$E_T(l,d) = \begin{cases} lE_{elec} + l\varepsilon_{fs}d^2 & \text{for} \quad d < d_0 \\ lE_{elec} + l\varepsilon_{mp}d^4 & \text{for} \quad d \geq d_0 \end{cases} \tag{3.1}$$

The energy required by the radio to receive an *l*-bit message is given by

$$E_R(l) = lE_{elec} \tag{3.2}$$

The E_{elec} depends on several factors such as digital coding, modulation, filtering, and spreading of the signal, whereas the amplifier energy, $\varepsilon_{fs}d^2/\varepsilon_{mp}d^4$, depends on the distance between the transmitter and the receiver and also on the acceptable bit-error rate. It should be noted that this is a simplified model. In general, radio wave propagation is highly variable and difficult to model.

3.3. Fault model

There are various reasons for the failure of WSNs. Firstly, sensor nodes may fail due to the depletion of their battery power, malfunctioning of hardware components (such as a processing unit and transceiver) or damage by an external event. Secondly, the wireless links of the WSNs may fail due to permanent or temporary blocked by an obstacle or environmental condition. The link failure causes the network partitions and dynamic changes in network topology. In this paper, we consider only permanent failure of the CHs and do not consider the transient and Byzantine failure of the WSNs. We assume the permanent failure of the CHs due to failure of the hardware components or complete energy depletion or permanent link failure. In this paper, we consider the reliability of the CHs following the Weibull distribution [23] which is extensively used for fault model as it can provide diverse failure patterns over time with its parameters. The probability density function of the Weibull distribution has the following form:

$$f(t) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta-1} e^{-\left(\frac{t}{\eta}\right)^{\beta}} \tag{3.3}$$

where β and η are the shape and the scale parameters respectively. This function indicates the likelihood of failure at time *t*. The reliability function of the Weibull distribution is given by

$$R(t) = e^{-\left(\frac{t}{\eta}\right)^{\beta}} \tag{3.4}$$

The reliability function is the probability that a device is functioning at time *t*. The failure rate depends on the shape parameter β as follows. When $\beta = 1$, the failure rate is constant; when $\beta > 1$, the failure rate increases over time and when $\beta < 1$, the failure rate decreases over time, i.e., the component strengthens or hardens over time. Note that $\beta > 1$ for wireless sensor networks which we consider in all the experiments.

4. Proposed algorithm

For better understanding of the proposed algorithms, we first describe some terminologies as follows.

(1) A set of sensor nodes denoted by $S = \{s_1, s_2, \ldots, s_n\}$.
(2) The set of gateways is denoted by $\xi = \{g_1, g_2, \ldots, g_m\}$ and g_{m+1} indicates the base station (BS), $n > m$.
(3) $d_{\max}$ denotes the maximum communication range of the gateways.

(4) $dis(s_i, s_j)$ denotes the distance between s_i and s_j.
(5) $E_{Residual}(s_i)$ denotes the remaining energy of s_i.
(6) $ComCH(s_i)$: The set of all those gateways, which are within the communication range (R_S) of sensor node s_i. In other words,

$$ComCH(s_i) = \{g_j | dis(s_i, g_j) \leq R_S \wedge g_j \in \xi\} \tag{4.1}$$

Therefore, s_i can be assigned to any one of the gateway from $ComCH(s_i)$, where $ComCH(s_i) \subseteq \xi$.
(7) $Neighbor(s_i)$ is the set of all those sensor nodes, which are within the communication range of node s_i. Therefore,

$$Neighbor(s_i) = \{s_j | dis(s_i, s_j) \leqslant R_S \wedge s_j \in \{S - s_i\}\} \tag{4.2}$$

$Com(g_i)$: The set of gateways, which are within communication range of g_i. The BS may also be a member of $Com(g_i)$. In other words,

$$Com(g_i) = \{g_j | \forall g_j \in \{\xi + g_{m+1}\} \wedge dis(g_i, g_j) < d_{\max}\} \tag{4.3}$$

(8) $HCount(g_i)$ denotes the number of next hops required to reach to the BS from g_i. If g_i can directly communicate with BS, then $HCount(g_i)$ is one. Therefore, $HCount(g_i)$ can be recursively defined as

$$HCount(g_i) = \begin{cases} 1, & \text{If next hop is} g_{m+1} (\text{i.e., BS.}) \\ 1 + HCount(g_j), & \text{next hop is} g_j (\text{i.e., other than BS.}) \end{cases} \tag{4.4}$$

(9) $BackCH(g_i)$: The set of gateways from $Com(g_i)$ with lesser hop count than g_i. Therefore,

$$BackCH(g_i) = \{g_k | \forall g_k \in Com(g_i) \, and \, HCount(g_i) < HCount(g_k)\} \tag{4.5}$$

Depending on the communication range and connectivity between the sensor nodes and the gateways, we have defined few kinds of sensor nodes in the system as follows [24].

Definition 1 (*Covered Node and Covered Set*). Covered nodes are those sensor nodes, which have at least one gateway within its communication range. Covered set is the set of all covered nodes in the WSN. We refer this set as 'CO_{set}'. It is obvious to note that a sensor node s_i belongs to CO_{set}, if it satisfies the following criteria:

$$s_i \in CO_{set} \iff [\{\exists g_j | g_j \in ComCH(s_i) \wedge g_j \in \xi\}] \tag{4.6}$$

Definition 2 (*Uncovered Node and Uncovered Set*). Uncovered nodes are those sensor nodes, which have no gateway within its communication range. Uncovered set is the collection of all uncovered nodes in the WSN. We refer this set as $UnCO_{set}$. A sensor node s_i belongs to $UnCO_{set}$, if it satisfies the following criteria:

$$s_i \in UnCO_{set} \iff [s_i \notin CO_{set}] \tag{4.7}$$

Definition 3 (*Backup nodes and Backup set*). Backup nodes of an uncovered sensor node s_i are all the covered sensor nodes which are within communication range of s_i. Backup set of an uncovered sensor node s_i is the set of all backup nodes of s_i. We refer this set as $BackupSet(s_i)$. Therefore,

$$BackupSet(s_i) = \{s_j | s_j \in Neighbor(s_i) \wedge s_j \in CO_{set}\} \tag{4.8}$$

Definition 4 (*Alive and Inactive Sensor node*). Alive sensor nodes are those sensor nodes, which have some residual energy and can send the sensed data to the CH directly or indirectly.

Sometimes, few sensor nodes may have residual energy, but unable to communicate with its CH due to CH failure. These sensor nodes are called inactive sensor nodes.

4.1. Bootstrapping

At the beginning, all the nodes undergo bootstrapping process, in which the BS broadcasts a *HELLO* message at a certain power level. By this way, each gateway can compute the approximate distance to the BS based on the received signal strength indicator [12,21]. It helps the gateways to select the proper power level to communicate with the BS. Then the BS floods a *HopPacket* which consists of a counter indicating the hop value. Initially the counter is set to one. After receiving the packet, a gateway stores the counter value, increments it by one and transmits it to their neighbor gateways. This transmission will continue until all the CHs are aware of their hop values for their use in data routing phase. Note that as the communication is through the wireless link, the gateways which have already transmitted their data packets, will also receive packets from their neighbor CHs. Therefore, after receiving the packet a gateway will check the counter value with its own stored one. If the received counter value is less than the stored value, then only the gateway increments the counter value, restores and retransmits it; otherwise it will ignore the received packets. This will help to avoid the redundant transmission and loop creation.

Network setup consists of a setup phase followed by the steady-state phase as shown in Fig. 2 and illustrated as follows. In the setup phase, clusters are formed using the proposed algorithm discussed later (Fig. 3). Steady-state phase divides the schedule of the network into multiple rounds of fixed duration. In each round, the gateways receive the sensed data from the cluster members and aggregate them to transfer it to the base station using the proposed distributed multi-hop routing algorithm discussed later. The steady-state phase is composed of some pre-specified rounds, say 50 or 75 rounds. Setup and steady-state phases are repeated until all the sensor nodes become inactive. We now present the details of cluster formation and inter cluster routing in the following sections subsequently.

4.2. Distributed clustering algorithm

In the setup phase, gateways broadcast a HELLO message within the communication range of a sensor node. The HELLO message consists of gateway ID, residual energy and distance to the base station. After a certain timeout, if a sensor node s_i has received at least one HELLO message, then it becomes a member of CO_{set}, otherwise it becomes a member of $UnCO_{set}$. Each sensor node $S_i \in UnCO_{set}$, broadcasts a HELP message for backup. All the sensor nodes of $Neighbor(s_i)$ which are belonged to CO_{set}, send a reply to form $BackupSet(s_i)$. If the backup set is not empty (i.e., $|BackupSet(s_i)| \neq 0$), then s_i uses s_j from $BackupSet(s_i)$ as a *relay* with highest residual energy through which S_i could send the data to the CH. In other words, s_i uses s_j as a *relay* if

$$E_{Residual}(s_j) = Max\{E_{Residual}(s_k) | \forall s_k \in BackupSet(s_i)\} \tag{4.9}$$

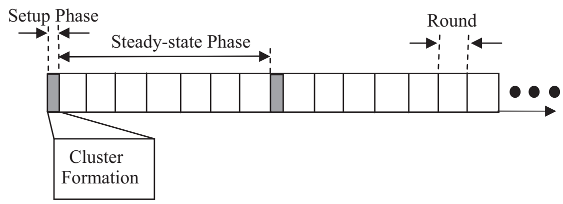

Fig. 2. Network setup.

Algorithm: Cluster-Formation

Step 1: $ComCH(s_i) = \{\}$;
$BackupSet\ (s_i) = \{\}$;
Step 2: ***while* (*not timeout*) *do***
if (s_i receives a HELLO message from g_k) ***then***
2.1: s_i becomes a member of CO_{set}
2.2: $ComCH\ (s_i) = ComCH\ (s_i) \cup g_k$
endif
endwhile
Step 3: ***if*** (no HELLO message is received by s_i) ***then***
s_i becomes a member of $UnCO_{set}$.
endif
Step 4: ***if*** ($s_i \in UnCO_{set}$) ***then***
4.1: s_i broadcasts a HELP message for backup.
4.2: ***while*** (s_i receiving reply from s_j)
$BackupSet(s_i) = BackupSet(s_i) \cup s_j$
endwhile
4.3: ***if*** ($|BackupSet\ (s_i)| \neq 0$) ***then***
s_i uses s_j from $BackupSet\ (s_i)$ as a *relay* with highest residual energy to send the data to the CH.
endif
else
4.4: s_i calculates cost of all these CHs according to equation 4.13 and joins the CH with the highest cost value.
4.5: ***if*** (s_i receives any HELP message) ***then***
Send reply message.
endif
endif
Step 5: ***Stop.***

Fig. 3. Cluster formation algorithm.

Thus the sensor nodes, which are not covered due to random deployment or sudden failure of CHs, are assigned to a CH through multi-hop communication. Now, the sensor nodes s_j, $\forall s_j \in CO_{set}$, join a CH based on the cost value. Let $CHCost(g_i, s_j)$ be the cost of the cluster head g_i for s_j. To define the cost function, we consider the following parameters.

(1) *Residual Energy of CH*: Sensor node should join that CH which has higher residual energy than any other CH within its cluster range. Therefore,

$$CHCost(g_i, s_j) \propto E_{\text{Residual}}(g_i) \tag{4.10}$$

(2) *Distance from Sensor node to CH*: As a non-CH sensor nodes consume maximum energy to communicate with its CH, sensor node should join the nearest CH. The shorter the distance higher is the chance to join. Therefore,

$$CHCost(g_i, s_j) \propto \frac{1}{dis(s_j, g_i)} \tag{4.11}$$

(3) *Distance from CH to Base Station*: Gateways are capable of long-haul communication compared to sensor nodes and can communicate direct with the BS. Thus the gateways which are far away from the BS, consume more energy for long-haul communication to the BS. Therefore, cluster member of these CHs should be less than that of the CHs which are nearer to the BS. In other words,

$$CHCost(g_i, s_j) \propto \frac{1}{dis(g_i, g_{m+1})} \tag{4.12}$$

By combining Eqs. (4.10), (4.11) and (4.12), we obtain

$$CHCost(g_i, s_j) \propto \frac{E_{\text{Residual}}(g_i)}{dis(s_j, g_i) \times dis(g_i, g_{m+1})}$$

$$\text{i.e., } CHCost(g_i, s_j) = K \times \frac{E_{\text{Residual}}(g_i)}{dis(s_j, g_i) \times dis(g_i, g_{m+1})}$$

where K is a proportionality constant. With the similar reason, we take the weight value of the sensor nodes only for comparison purpose. So, the value of K does not hamper our objective and without loss of generality, we can assume $K = 1$. Therefore,

$$CHCost(g_i, s_j) = \frac{E_{\text{Residual}}(g_i)}{dis(s_j, g_i) \times dis(g_i, g_{m+1})} \tag{4.13}$$

Now S_j selects the cluster head say G_i with highest cost value. Therefore,

$$CHCost(g_i, s_j) = Max\{CHCost(g_k, s_j) | \forall g_k \in ComCH(s_j)\} \tag{4.14}$$

The sensor nodes (s_j, $\forall s_j \in CO_{set}$) select a CH to join using Eq. (4.14) and then sends a JOIN_REQ message to its selected CH using a non-persistent CSMA/CA MAC protocol. After completion of cluster formation, each CH sets up a TDMA schedule for its member nodes and transmits this schedule to the member sensor nodes. This TDMA schedule ensures that no collisions will occur during communication between sensor nodes and CH. It also ensures that the radio component of each sensor node will be turned off except their transmit time. Thus, it reduces the energy consumption. The cluster formation algorithm for sensor node s_i is given in Fig. 3. This algorithm simultaneously runs by each sensor node.

Lemma 4.1. *The Cluster-Formation algorithm has worst case time complexity of $O(n)$ where n is the number of sensor nodes in the network.*

Proof. Step 1 and Step 3 of the algorithm takes constant time. Step 2 can be executed in $O(m)$ time as a sensor node may have m number of gateways within its communication range in the worst case. In the Step 4, sensor node resets its backup set (Step 4.2) in $O(n)$ time and selects a CH in $O(m)$ processing time (Step 4.4). The overall execution time of Step 4 is $O(m) + O(n)$, i.e., $O(n)$ in a worst case situation as $n > m$. Therefore, the overall processing time of the cluster formation algorithm is $O(n)$ in the worst case as Step 4 dominates all other steps of the algorithm. □

Lemma 4.2. *The Cluster-Formation algorithm has worst case message exchange complexity of $O(1)$ per sensor node, i.e., $O(n)$ for the whole network having n sensor nodes.*

Proof. In cluster formation phase, a sensor node calculates the cost value of the CHs within its communication range and sends a join request to the selected CH. However, the sensor nodes which have no CHs within their communication range, broadcast a HELP request message (step 4.1). If it finds any sensor node with helping hand, then it sends a join request message to join that cluster via the helping sensor node with multi-hop communication. Therefore, sensor node has to send only two messages in the worst case for the formation of the cluster. Hence, message complexity of each sensor node is constant, i.e., $O(1)$. Thus, overall message exchange complexity for the whole network is $O(n)$. □

4.2.1. Fault tolerance

During the steady state phase, at any time a CH may fail due to depletion of energy or some damages. The fault can be detected whenever the member sensor nodes do not receive any data acknowledgment receipt from the CH and also the failure of the CH can be confirmed from the neighbor sensor nodes of the same cluster [25]. After detection of the fault, the member sensor nodes of the faulty cluster broadcast a HELP message within its communication range. The sensor nodes s_j, $\forall s_j \in Neighbor(s_i)$, from other cluster and the CHs g_k, $\forall g_k \in ComCH(s_i)$, reply this HELP message. If the sensor node receives any reply from other CHs then it remains as elements of CO_{set}, otherwise it becomes the elements of $UnCO_{set}$. The sensor nodes s_i, $\forall s_i \in UnCO_{set}$, those have received reply from other sensor nodes reset their $BackupSet(s_i)$. Now, s_j, $\forall s_j \in CO_{set}$, joins a CH considering the cost value as in Eq. (4.13) and s_i, $\forall s_i \in UnCO_{set}$, uses s_j, $s_j \in BackupSet(s_i)$, as a *relay* with highest residual energy to send the data to the CH. The fault tolerance algorithm for sensor node s_i is given in Fig. 4. This algorithm is individually run by the sensor nodes from the faulty cluster.

4.3. Distributed routing

We now present the proposed routing algorithm. Here, gateways select the next-hop relay nodes in such a way that it requires less energy and also takes care balancing of energy consumption of the gateways (CHs). This is implemented as follows. The gateways are aware of their distance from the BS and also the hop count as discussed in bootstrapping (refer Section 4.1 and also [12,22]. As the communication is through a wireless link, gateways are also aware of some information about their neighbor gateways such as the distance from their neighbors, neighbor distance from the BS, neighbor remaining energy and hop count as in [12,22]. Generally a gateway may have many possible next-hop relay nodes within its communication range. However, in our approach only those neighbor gateways are eligible to serve as a relay node which has lesser hop count. Now, a gateway g_j chooses the next hop relay node g_r from the eligible nodes such that $HCount(g_i) > HCount(g_r)$) based on the cost function. We denote the cost function as $Cost(g_j, g_r)$ which indicates the cost of selecting the gateway g_r as next hop relay node by the gateway g_j. For energy balancing of the gateways during data routing, our method bases on the number of those CHs which can select a particular CH (say g_r) as a next hop relay node. We call all such CHs as backward CHs denoted by $BackCH(g_r)$ (refer Eq. (4.5)). Note that the CHs having many backward CHs have high possible data forwarding load. The cost function for the same is derived similarly as Eq. (4.13) as follows.

$$Cost(g_j, g_r) = \frac{E_{Residual}(g_r)}{dis(g_j, g_r) \times dis(g_r, g_{m+1}) \times |BackCH(g_r)|} \tag{4.15}$$

Now, g_j selects the gateway g_k having maximum cost value as relay node. Therefore,

$$Cost(g_j, g_k) = Max\{Cost(g_j, g_r) | \forall g_r \in Com(g_j)\, and\, HCount(g_j) > HCount(g_r)\} \tag{4.16}$$

Note that the CHs with hop count 1 sends their data directly to the base station without execution of the next hop selection algorithm. Other CHs forward the data to the next-hop relay node using a non-persistent CSMA/CA MAC protocol. The next hop relay node selection algorithm for g_i is given in Fig. 5.

Algorithm: Clustering-Fault-Tolerance

Step 1: S_i broadcast a HELP message within its communication range.
  1.1: $ComCH(s_i) = \{\}$;
  1.2: $BackupSet\,(s_i) = \{\}$;
Step 2: ***while*** (s_i receiving reply)
  if(reply is from gateway g_k)
    2.1: s_i becomes the elements of CO_{set}
    2.2: $ComCH\,(s_i) = ComCH\,(s_i) \cup g_k$
  else if (reply is from sensor node s_j)
    $BackupSet\,(s_i) = BackupSet\,(s_i) \cup s_j$
    endif
  endif
endwhile
Step 3: ***if*** $(s_i \in UnCO_{set}\ \&\&\ |BackupSet(s_i)| \neq 0)$ ***then***
  s_i uses s_j from $BackupSet\,(s_i)$ as a *relay* with highest residual energy to send the data to the CH.
else
  s_i calculates cost of all CHs from $ComCH(s_i)$ according to equation 4.13 and joins the CH with the highest cost value.
endif
Step 4: ***Stop.***

Fig. 4. Fault tolerant clustering algorithm.

Algorithm: *Next-hop-Selection*

```
Input: 1) HCount(g_i)
       2) ∀g_k ∈ Com(g_i): Residual energy, hop count, distance to g_k and distance from
          g_k to BS.
Output: Next-hop relay node (Relay) of g_i.
-------------------------------------------------------------------------
Step 1: If (HCount(g_i)==1) then
           1.1: Relay= g_{m+1} /* g_{m+1} indicates BS.*/
        else
           1.2: T = Com(g_i)
           1.3: max = -1.0
           1.4: Relay = -1
           1.5: While (T ≠ Φ) do   /*  Φ = NULL */
                   Select a g_r from T
                   If(Cost(g_i, g_r) > max && HCount(g_i) > HCount(g_r)) then
                      Relay = g_r
                      max = Cost(g_i, g_r)
                   endif
                   T = T - g_r
                endwhile
           1.6: If(Relay == -1 && HCount(g_i) ≤ m) then
                   HCount(g_i) = HCount(g_i) +1
                   Repeat Step 1.2 to Step 1.5
                endif
        endif
Step 2: Stop.
```

Fig. 5. Next-hop selection algorithm.

Remark 4.1. Note that the above next-hop selection algorithm has the ability to cope up with the failure of the CHs as follows. Whenever a CH does not receive any acknowledgment from its neighbor CH, it detects the neighbor as failed CH. This can also be confirmed by asking the other neighbor CHs. Note that failure of one or more CHs does not create any problem in data routing phase if there is still a live CH with less hop count to serve as a relay node. But, there may be a situation when a CH may not have any next-hop CH with lower hop count. Then the CH increases its hop count by one and broadcasts the same to its neighbors to update their hop count. Now the CH searches for other CHs having lesser hop count than itself. If it finds some CHs with the lesser hop count, then it selects one of them as a next hop relay node considering the cost value as described in Eq. (4.15).

Lemma 4.3. *The Next-hop-Selection algorithm has worst case time complexity of $O(m)$ without any fault and $O(m^2)$ with occurring of faults where m is the number of gateways.*

Proof. Step 1.1 through step 1.4, require constant time. To select a next hop relay node, a gateway has to compare the cost value of its neighbor gateways in step 1.5 which takes $O(m)$ time, assuming that a gateway may have m-1 neighbor gateways in worst situation. Therefore, next hop selection takes $O(m)$ time in the worst case. However, whenever fault occurs then a gateway increases its hop count by 1 and step 1.2 through step 1.5 are repeated requiring $O(m^2)$ time as a gateway can increase its hop count a maximum of m-1 times. □

Lemma 4.4. *The routing graph of the network is loop-free in the Next-hop-Selection algorithm.*

Proof. Let us assume that there exists a loop $g_a \rightarrow g_b \rightarrow g_c \ldots g_r \rightarrow g_a$ in the routing graph. As per the algorithm, a CH g_i selects g_k $(i \neq k)$ as a next hop relay node only when it fulfils the condition $HCount(g_i) > HCount(g_k)$. Therefore, if we consider the graph up to the point when loop is not still created then we find that $HCount(g_a) > HCount(g_b) > HCount(g_c) > \ldots > HCount(g_r)$. As per the assumption, a loop is created because g_r selects g_a as its next hop relay node, which implies that $HCount(g_r) > HCount(g_a)$. But this contradicts $HCount(g_a) > HCount(g_b) > HCount(g_c) > \ldots > HCount(g_r)$. □

5. Experiment results

5.1. Experimental setup

We performed extensive experiments on the proposed algorithm using MATLAB R2012b and C programming 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. The experiments were performed with varied number of sensor nodes ranging from 400–500 and 40–50 gateways. Each sensor node was assumed to have an initial energy of 2 J and each gateway with 10 J. In the experiments, we considered the faults of the gateways following the Weibull reliability function with $\eta = 3200$ and $\beta = 3$. In the simulation run, we used the same parametric values as in [4,13] as shown in Table 1.

Table 1
Simulation parameters.

Parameter	Value
Sensor nodes	400–500
Gateways	40–50
Initial energy of sensor nodes	2.0 J
Number of simulation iterations	200
Sensor communication range	60 m
Gateway communication range	100 m
E_{elec}	50 nJ/bit
ε_{fs}	10 pJ/bit/m^2
ε_{mp}	0.0013 pJ/bit/m^4
d_0	30.0 m
E_{DA}	5 nJ/bit
Packet size	4000 bits
Message size	200 bits

For the sake of comparison, we also ran the fault tolerant clustering algorithm (FTCA) [9], the distributed algorithm DEBR [12] and MHRM [11]. Note that DEBR and MHRM are basically routing algorithms without any clustering phase. Therefore, we ran both these algorithms along with our proposed cluster formation algorithm for the fair comparison. We tested our proposed algorithms extensively and depict the experimental results for both the routing and clustering in a combined way. The simulation run was carried out with various network scenarios. However, due to the page limitation, we show the results of two different network scenarios, namely WSN#1 and WSN#2.

5.2. Experimental results for WSN#1

Here we consider the network scenario WSN#1 with sensing field of 400 × 400 square meters area and the position of the base station is taken at the boundary side of the region with coordinates (200, 200). We preformed the simulation run for 400 sensor nodes and 40 gateways.

5.2.1. Number of dead CHs and packets received (WSN#1)

Fig. 6(a) shows the comparison results in terms of number of dead CHs round by round. Note that the number of dead CHs includes all the CHs which are dead due to complete energy depletion or device failure following Weibull reliability function. It can be observed that DFCR performs better than the FTCA, DEBR and MHRM in terms of number of dead CH during the network life time. The rationale behind the poor performance of FTCA is that, the CHs use direct communication with BS which leads to initial death of the CHs due to huge energy consumption for long haul transmission. On the other hand, in order to balance the energy consumption some CHs in DEBR try to send the data packets to the base station through other CHs which may be in opposite direction of the base station and in the same way the packets finally may reach to a particular region from where there may not be any forward path reaching to the base station. As a result, a massive amount of energy is

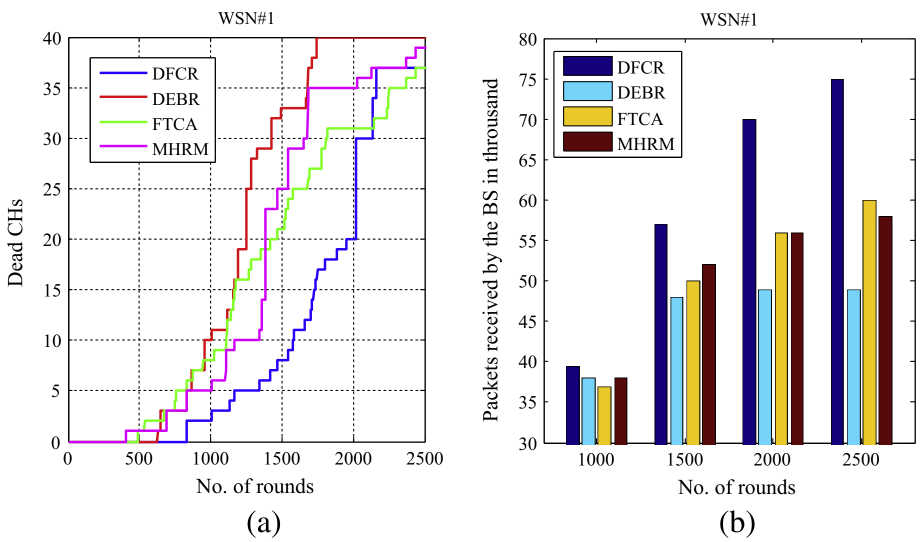

Fig. 6. Comparison in terms of (a) number of dead CHs and (b) number of packets received by the base station in WSN#1.

consumed without data packets reaching to the final destination, i.e., the BS. To minimize the hop count, the CHs in MHRM select longest distance CH towards the BS. As a result, the CHs die quickly for huge energy consumption due to long haul communication. Whereas, DFCR prevents the initial death of the CHs through properly balance the energy consumption and energy efficient selection of next hop CHs. Moreover, if a few CHs die due to complete energy depletion or device failure, DFCR finds a route to the BS until there exists a route and the network becomes connected for longer period of time. Note that the long life of the CHs helps to reach more number of data packets to the BS. We also compare the performance of the algorithms in terms of total data packets sent to the BS until the last gateway becomes inactive or disconnected. The DFCR performs better than the existing algorithms in terms of the number of packets received by the BS as evident from Fig. 6(b).

5.2.2. Total energy consumption and average energy consumption (WSN#1)

Fig. 7(a) shows that total energy consumption by the networks for the algorithms. Note that although, the rate of consumption of energy is almost same for all the algorithms, DFCR can be shown to perform better than the others as follows. As the number of live CHs in DFCR is higher than the others, it transmits higher number of data packets to the BS which generally consumes comparatively more energy. The average energy consumption per packet in DFCR is comparably lower than the others as shown in Fig. 7(b) which also indicates the energy efficiency quality of the proposed work.

5.2.3. Standard deviation and number of inactive sensor nodes (WSN#1)

We next compare the algorithms in terms of energy balancing of the CHs. In order to judge this quality, we calculated the standard deviation of the CH's residual energy and plotted against the number of rounds. The standard deviation of the CH's residual energy indicates even distribution of energy. This is noteworthy from the Fig. 8(a) that the standard deviation of remaining energy of CHs in DFCR is lower than the others, that is more energy balancing due to the derived cost function.

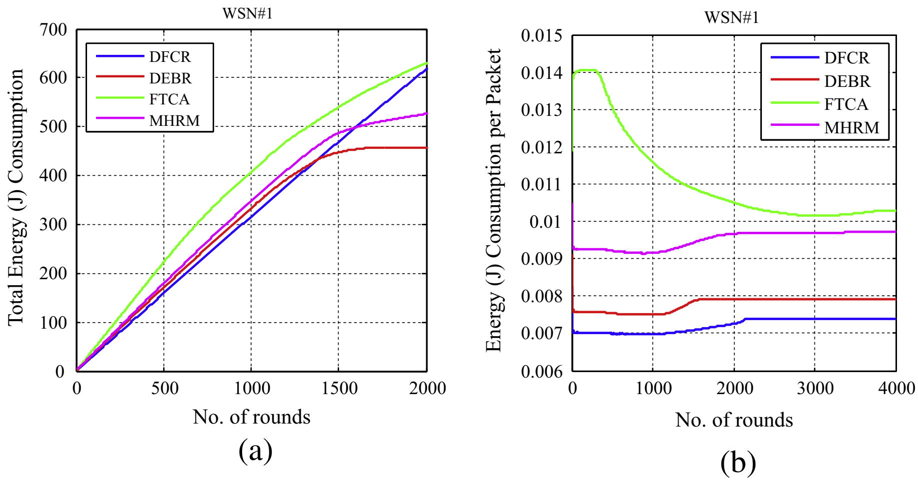

Fig. 7. Comparison in terms of (a) total energy (J) consumption by the network and (b) average energy consumption per packet in WSN#1.

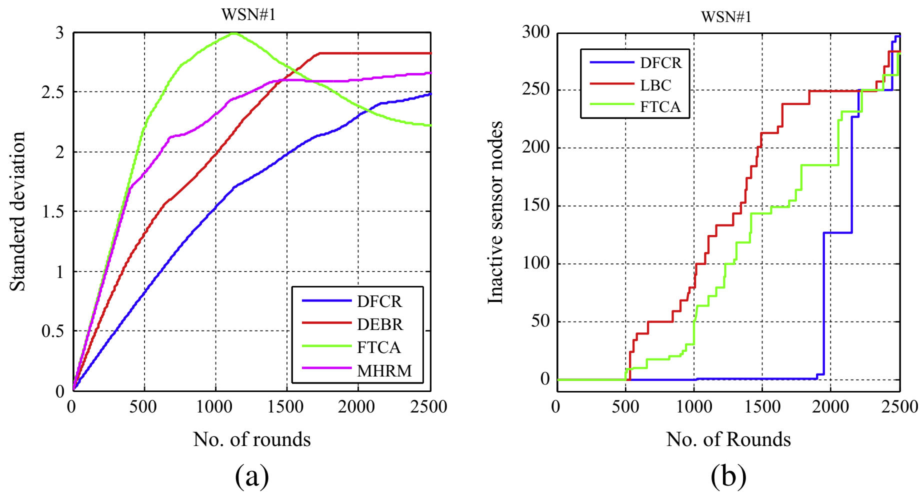

Fig. 8. Comparison in terms of (a) standard deviation of remaining energy of CHs and (b) number of inactive sensor nodes in WSN#2.

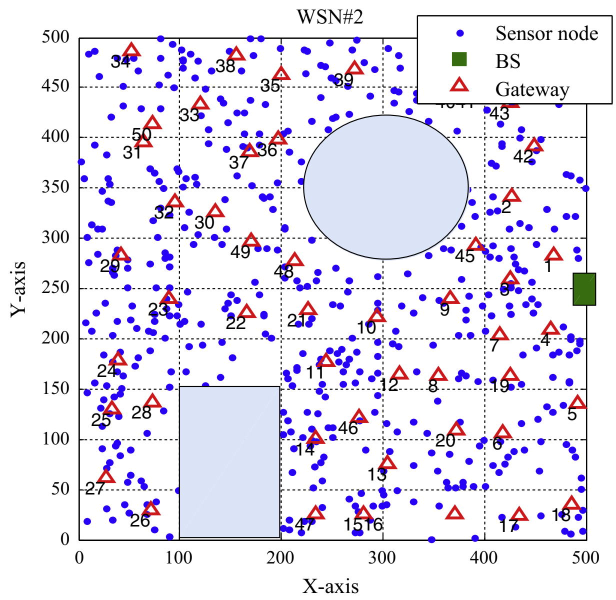

Fig. 9. The network scenario of WSN#2 with randomly deployed 500 sensor nodes and 50 gateways.

Here, we compare the algorithms in terms of number of inactive sensor nodes in the network. A sensor node is considered as active if its existing energy is not zero and also there must be at least one gateway within its communication range. Sometimes few gateways die quickly for improper energy balancing. As a result, few sensor nodes are unable to find any gateway within their range, though it still may have some existing energy. In our scenario, this type of sensor node is also considered as inactive. As DEBR and MHRM consider only traditional routing, for the sake of simulation we compare the proposed algorithm with the clustering algorithms FTCA and LBCA [6]. It can be observed from Fig. 8(b) that the number of inactive sensor nodes round by round in DFCR is increasing lesser than the existing algorithms. This is due to the fact that DFCR takes care about the energy consumption of the normal sensor nodes. Note that LBCA only balances the load of the CHs. To achieve this goal, some sensor nodes are assigned to the CH which may be farther from it. As a result their energies are drained out due to long haul transmission and die quickly.

5.3. Experimental results for WSN#2

We also consider another scenario (WSN#2) with sensing field of 500 × 500 square meters area and the position of the base station is taken at a boundary side of the region, i.e., at the location (500, 250). Here, we assumed a real life complex network scenario where few region of the sensing field may be vacant and analyzed the performance of the algorithms. Here also we consider the failure of the CHs due to complete energy depletion or device failure using Weibull reliability function.

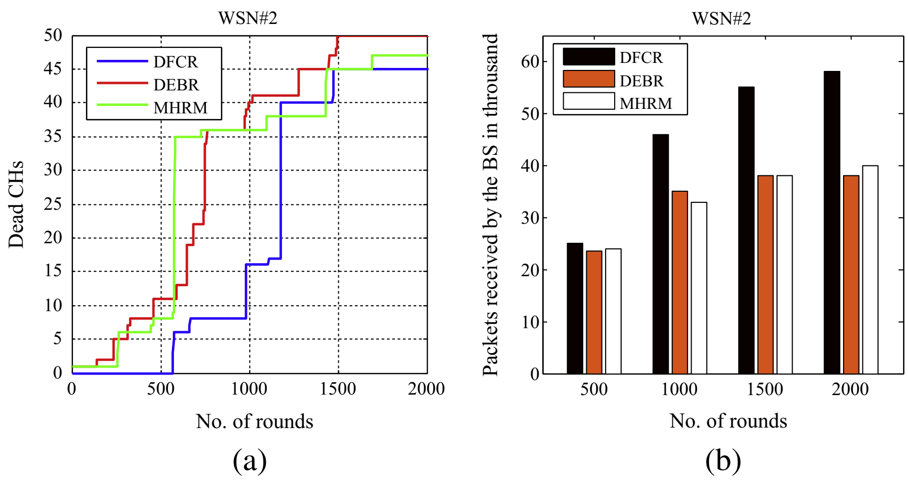

Fig. 10. Comparison in terms of (a) number of dead CHs and (b) number of packets received by the base station in WSN#2.

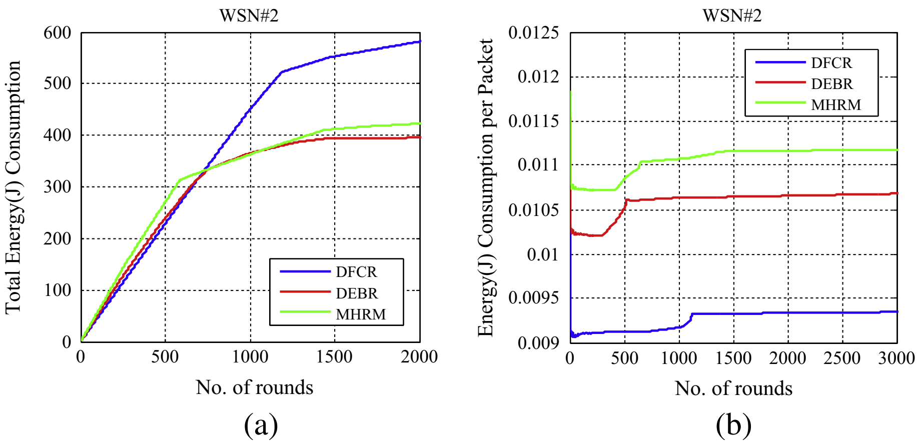

Fig. 11. Comparison in terms of (a) total energy (J) consumption by the network and (b) average energy consumption per packet in WSN#2.

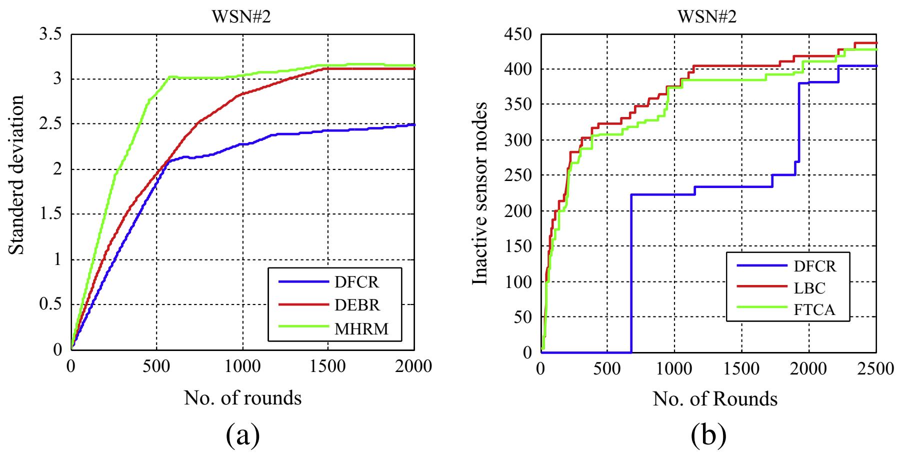

Fig. 12. Comparison in terms of (a) standard deviation of remaining energy of CHs and (b) number of inactive sensor nodes in WSN#2.

We created two vacant areas in the sensing field as follows. A rectangular area of 100 × 150 square meters and a circular area of radius 80 m as shown in Fig. 9. Here, we preformed the simulation for 500 sensor nodes and 50 gateways. Note that FTCA [9] considers direct communication from the CHs to the base station which may not be applicable in a large area network like WSN#2. Therefore, we compare the proposed algorithm DFCR with DEBR and MHRM only.

We first show the comparison results in terms of number of dead CHs in Fig. 10(a). The proposed algorithm performs comparably better in this scenario too. Fig. 10(b) also shows the better performance of DFCR than the existing algorithms in terms of total number of data packets received by the BS.

Fig. 11(a) shows that total energy consumption by the networks which is almost same as it was in WSN#1. But, Fig. 11(b) shows the average energy consumption per in DFCR packet is significantly lower that the existing algorithms. We also show the performance of the algorithms in terms of energy balancing of the in Fig. 12(a) which shows that the standard deviation of remaining energy of CHs per round is comparably lower than the others. Fig. 12(b) shows a lower number of inactive sensor nodes in DFCR than the LBCA and FTCA in the complex scenario too.

6. Conclusion

Energy conservation of cluster heads is central to design of clustering and routing algorithms for a large scale wireless sensor network. Furthermore, sensor nodes are very prone to failure in many applications. This paper has presented a distributed clustering and routing algorithm called DFCR for wireless sensor networks which is energy efficient as well as fault tolerant. The cluster formation is based on residual energy of the CHs, distance between the member sensor nodes to their CHs and also the distance from the CHs to the base station. The clustering algorithm has been shown to require $O(n)$ time complexity for n sensor nodes and $O(1)$ message exchange complexity per sensor node in the worst situation. The routing algorithm has been shown to run in $O(m)$ time without any fault and $O(m^2)$ time with occurrence of faults in the worst case. The routing algorithm has also been shown to be loop free and to cope with sudden failure of cluster heads in the data routing path. We have presented various experimental results using two different scenarios of the WSN. It has been shown that the proposed algorithm outperforms the existing algorithms, namely FTCA, MHRM and DEBR in terms of number of dead CHs, energy consumption, number of data packets received by the base station. The algorithm has also been shown to be more efficient than the algorithms FTCA and LBCA in terms of number of inactive sensor nodes. However, for fault tolerance of the proposed algorithms, we have considered only permanent failure of the CHs. In future, our attempt will be made to design energy aware distributed clustering and routing algorithm emphasizing partial and transient failure of the sensor nodes.

Acknowledgments

The first version of the paper was appeared in the proceedings of the International conference ICACCI 2013 (IEEE Xplore, pp. 997–1002). The authors are thankful to the anonymous reviewers for their valuable comments.

References

[1] Akyildiz IF, Su W, Sankarasubramaniam Y. Wireless sensor networks: a survey. Comput Netw 2002;38:393–422.
[2] Abbasi AH, Younis M. A survey on clustering algorithms for wireless sensor networks. Comput Commun 2007;30:2826–41.
[3] Akkaya K, Younis M. A survey on routing protocols for wireless sensor networks. Ad Hoc Netw 2005;3:325–49.
[4] Kuila P, Gupta SK, Jana PK. A novel evolutionary approach for load balanced clustering problem for wireless sensor networks. Swarm Evol Comput 2013;12:48–56.
[5] Bari A et al. Design of fault tolerant wireless sensor networks satisfying survivability and lifetime requirements. Comput Commun 2012;35(3):320–33.
[6] Low CP et al. Efficient load-balanced clustering algorithms for wireless sensor networks. Comput Commun 2008;31(4):750–9.
[7] Kuila P, Jana PK. Improved load balanced clustering algorithm for wireless sensor networks. In: Proc. of int. conf. ADCONS 2011, LNCS 7135; 2012. p. 399–404.
[8] Kuila P, Jana PK. Approximation schemes for load balanced clustering in wireless sensor networks. J Supercomput 2014;68(1):87–105.
[9] Gupta G, Younis M. Fault-tolerant clustering of wireless sensor networks. Proc Int Conf IEEE WCNC 2003;3:1579–84.
[10] Kuila P, Jana PK. Energy efficient load-balanced clustering algorithm for wireless Sensor Network. Proc Technol 2012;6:771–7.
[11] Chiang SS et al. A minimum hop routing protocol for home security systems using wireless sensor networks. IEEE Trans Consum Electron 2007;53(4):1483–9.
[12] Ok Chang-Soo et al. Distributed energy balanced routing for wireless sensor networks. Comput Ind Eng 2009;57:125–35.
[13] Heinzelman W, Chandrakasan A, Balakrishnan H. Application specific protocol architecture for wireless microsensor networks. IEEE Trans Wireless Commun 2002;1(4):660–70.
[14] Tyagi S, Kumar N. A systematic review on clustering and routing techniques based upon LEACH protocol for wireless sensor networks. J Netw Comput Appl 2013;36:623–45.
[15] Kuila P, Jana PK. An energy balanced distributed clustering and routing algorithm for wireless sensor networks. In: Proc. OF INT. CONF. PDGC 2012, IEEE Xplore; 2012. p. 220–225.
[16] Intanagonwiwat C, Govindan R, Estrin D. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In: ACM intl. conf. on mobile computing and networking; 2000. p. 56–67.
[17] Djukic P, Valaee S. Reliable packet transmissions in multipath routed wireless net-works. IEEE Trans Mobile Comput 2006;5:548–59.
[18] Deb B, Bhatnagar S, Nath B. ReInForm: reliable information forwarding using multiple paths in sensor networks. In: Proceed of the 28th IEEE int. conf. on local computer networks; 2003. p. 406–15.
[19] Younis M et al. Topology management techniques for tolerating node failures in wireless sensor networks: a survey. Comput Netw 2013. 10.1016/j.comnet.2013.08.021.
[20] Lai Y, Chen H. Energy-efficient fault-tolerant mechanism for clustered wireless sensor networks. IEEE ICCCN 2007:272–7.
[21] Xu J et al. Distance measurement model based on rssi in wsn. Wireless Sensor Netw 2010;2:606–11.
[22] Baronti P et al. Wireless sensor networks: a survey on the state of the art and the 802.15.4 and ZigBee standards. Comput Commun 2007;30:1655–95.
[23] Lee Jae-Joon et al. Aging analysis in large-scale wireless sensor networks. Ad Hoc Netw 2008;6(7):1117–33.
[24] Azharuddin Md, Kuila P, Jana PK. A distributed fault tolerant clustering algorithm for wireless sensor networks. In: Proc. of ICACCI 2013, IEEE Xplore; 2013. p. 997–1002.
[25] Banerjee I et al. Effective fault detection and routing scheme for wireless sensor networks. Comput Electr Eng 2014;40:291–306.

Md Azharuddin received B. Sc and MCA from Aligarh Muslim University in 2008 and 2011 respectively. Currently he is a research scholar in the department of computer Science and Engineering, Indian School of Mines and pursuing his Ph. D work. His main research interests include wireless sensor networks.

Pratyay Kuila received B.Tech. and M.Tech. in 2008 and 2011 respectively both in Computer Science and Engineering from West Bengal University of Technology, Kolkata. His main research interest includes developing clustering and routing algorithms for Wireless Sensor Networks with the issue of energy efficiency and application of evolutionary algorithms.

Prasanta K. Jana received M. Tech. in Computer Science from University of Calcutta and Ph. D. from Jadavpur University in 1988 and 2000 respectively. Currently he is a Professor in Computer Science and Engineering department, Indian School of Mines, India. Jana is an IEEE senior member. His current research interest includes wireless sensor networks, parallel & distributed processing and cloud computing.