Peer-to-Peer Networking and Applications (2022) 15:45–55
https://doi.org/10.1007/s12083-021-01226-y

A novel hierarchical fault management framework for wireless sensor networks: HFMF

Elham Moridi[3] · Majid Haghparast[4] · Mehdi Hosseinzadeh[1,2] · Somayyeh Jafarali Jassbi[3]

Received: 4 January 2021 / Accepted: 19 July 2021 / Published online: 12 August 2021

Abstract
Wireless sensor nodes (WSNs) are employed to collect data for control and supervisory purposes in inaccessible areas. Applying sensors in inaccessible areas and their hardware limitations result in occurring faults and non-renewability of energy. Thus networks need a fault tolerant method to continue their optimal activity in the presence of faults. Here, through improving energy consumption and fault management, we propose a new hierarchical fault management framework to overcome the limitations. The proposed method complies with clustering algorithms. Hence, due to the importance of cluster head nodes a backup is employed to replace faulty ones. Also, data correlation among cluster members is used to cellularize the cluster nodes virtually. In this process, a cell remains in active mode as the representative of cell, and others are in sleep mode as spare ones. The purpose of this mechanism is to reduce the number of active cluster nodes and detect intermittent faults. To detect the permanent faults of nodes, self-detection method has been used. In addition, the proposed framework diagnoses and recovers faults in communication links between nodes. The results of simulation reveal that the proposed framework leads to improved energy consumption, alive nodes, and fault detection accuracy compared with other frameworks.

Keywords Wireless sensor nodes · Clustering · Fault detection · Fault recovery · Fault management framework

1 Introduction

In recent years, technology development has resulted in making small and relatively cheap sensor nodes, connected through a wireless network [1]. In WSNs, each node is consisted of a sensor unit, process unit, transmission unit, and power unit. The nodes receive environmental data through sensor unit, process it moderately, and transmit it to other nodes. Finally, the data is transferred to BS. The power for all these components is provided by a battery. The energy of the battery is limited, and the battery cannot be recharged or replaced [2, 3]. Due to the importance of energy, data reduction, energy-efficient routing protocols, and duty cycling are applied. The focus of data reduction methods is on reducing generated, processed, and transferred data; some of the most important methods are sampling-based and data correlation methods [4].

Energy is not the only challenge in WSNs. In fact, some factors such as employing WSNs in inaccessible areas and applying limitations in producing nodes to reduce costs cause them being prone to faults [5, 6].

WSNs consists of many sensor nodes, widely scattered in a harsh environment and collect data. The location of nodes is not necessarily predetermined and specified [7]. Since the nodes of network are deployed in harsh environments, fault is a major challenge for WSNs. Although there are various classifications for their faults, generally the faults can be classified at three levels: node, network, and sink or base station (BS). Nodes and BS may experience software or hardware faults; for example, they may transfer faulty data due to reduced battery energy. Also, some faults may be experienced at network level such as faulty transmission paths or links. Because one of the most Thus fault tolerance is a required quality of WSNs [8]. This quality enables them

✉ Majid Haghparast
haghparast@srbiau.ac.ir

1 Mental Health Research Center, Psychosocial Health Research Institute, Iran University of Medical Sciences, Tehran, Iran

2 Computer Science, University of Human Development, Sulaimaniyah, Iraq

3 Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

4 Department of Computer Engineering, Yadegar-E-Imam Khomeini (RAH) Shahre Rey Branch, Islamic Azad University, Tehran, Iran

to continue their optimal performance in spite of any faults in their components [9]. The main phases of fault tolerance in WSNs are FD (FD) and fault recovery (FR), which are defined as fault management (FM). A set of functions that can be applied to make other algorithms fault tolerant is called fault management framework (FMF).

FD means detecting any unpredictable failure or destructive factors that impact the optimal state of a network or node [10]. Different methods have been suggested for FD, the first phase of fault management. Based on the complexity of implementing these methods, FD methods are divided into 3 groups: calculation-based, protocol-based, and hybrid. FR is performed based on redundancy including node and path redundancy. In node redundancy, a node replaces the faulty node and in path redundancy, a new path replaces the faulty path.

In the proposed method, due to the importance of energy in sensor networks, we have combined energy management and fault management. Sleep/active method is used for energy management. In the proposed method for implementing fault management, in the first step after clustering the nodes, a spare node was selected for the cluster head. This is done with the aim of recovering the faults in the cluster heads. Then, in the second step of the proposed method, the nodes of the cluster members are placed in virtual cells. The purpose of virtual cells is to combine energy management and fault management methods. Then, by evaluating the data correlation of nodes, transient and intermittent faults can be detected. In addition, permanent hardware faults in nodes can be identified by self-detection methods. All hardware faults including battery, sensor unit, processing unit, transmitter and receiver circuit are detected. In addition to the hardware faults, the communication link fault between the nodes is also detected. In the last step, fault recovery is implemented in cluster head nodes and links.

2 Contribution

Regarding the challenge of energy and FM in WSNs, the purpose of this paper is to suggest a hierarchical fault management framework (HFMF) complying with improving energy consumption. The main contributions of the paper are summarized, as follows:

1. By combining fault management and energy management methods, we have proposed a fault management framework for clustering-based algorithms in WSNs. The proposed method can be implemented for all hierarchical clustering algorithms.

2. In HFMF, due to the importance of cluster heads node in clustering algorithms, a spare node is considered in the first step of fault management. By selecting a spare node for *CHs*, we increase the accuracy of fault detection and reduce delay recovery faults.

3. We detected transient and intermittent faults of cluster member nodes using data correlation. In the same step, we divided the nodes into virtual cells. At this point, we placed the cluster member nodes in the virtual cells.

4. We implemented the sleep/active method in each cell. Therefore, we have used this energy management method to reduce the number of active nodes in the network and decrease the possibility of faults in them. Therefore, as shown in the evaluation section, the number of live nodes in the proposed method is more than the compared frameworks.

5. In the HFMF method, we used the self-detection method to detect permanent faults in nodes. Since battery, sensor circuit, transceiver circuit, and processor are common to sensor nodes, all fault detection and recovery methods have been addressed in the proposed fault detection framework. As shown in the simulation section, the use of several fault detection methods in the proposed method has increased the accuracy of fault detection.

6. In the recovery step of the proposed HFMF method, we use the sleeping nodes as spare nodes. The node that was selected as a spare is also used to recover the fault in *CHs*. Therefore, in the proposed method, faults are managed in all network nodes (cluster heads and cluster members).

7. In this method, we manage the fault that occurred in the communication link. The faults of the communication links between nodes can be diagnosed through propagation speed, reliability of link, and lack of receiving ACKs.

In the proposed framework, the intermittent and permanent faults of sensors and communication links are detected. The data correlation between nodes is applied to detect intermittent faults. Also, permanent faults among nodes are detected through self-detection, which is managed by cluster heads. In the proposed method, some spare nodes have been selected as active and sleep to reduce delayed FR in *CHs* and *RNs*. The rest of the paper is organized as follows. Section 2 represents an overview of the related work. Then Sect. 3 presents models in the proposed framework. Section 4 includes the steps of the framework, and Sect. 5 provides the performance evaluation. Finally, Sect. 6 concludes this paper.

3 Related work

These days, several FMFs have been suggested for WSNs. These frameworks are categorized into 3 groups based on their implementation structures: centralized, distributed, and hierarchical. In centralized FMF, a centralized node identifies the geographical area of faulty nodes in the whole network [11, 12]. In Distributed FMFs, several managers are distributed to network to manage faults. Each manager controls one subset of network and can directly communicate with other management stations [13]. Hierarchical FMFs are a combination of centralized and distributed frameworks. The most important hierarchical frameworks in WSNs are as follows.

The framework suggested in [14] (CRAFT) is one of the most important hierarchical frameworks. The general idea of CRAFT is using checkpoint. In FD phase, BS discovers faulty *CHs* through response time expiry and monitoring residual energy. If the *CH* does not send any data to BS during expected time, its fault will be diagnosed. FMFs are suggested in [15] to improve CRAFT. In these frameworks, each *CH* is supposed to select a checkpoint from its cluster members. In these frameworks, *CHs* send aggregated data to the sink and save a copy of it at their checkpoint as well. In [16, 17], when *CH* does not send any responses to BS in a time slot, it is detected as a faulty *CH* so that the information is disseminated in the rest of network, and FR initiates. To recover the faulty *CH*, the sink will choose a cluster member as a new *CH*. In the FMF suggested in [18], *CH* maintains a timer for each node. When *CH* notices the performance of a node, it resets the timer for the node. If *CH* receives no responses from neighboring nodes before timer expiry or after sending 3 messages in a random time, the node will be considered dead.

In [19], a comprehensive fault tolerant framework has been explored. The general idea of the framework is to diagnose and recover faulty *CHs* and cluster members through an effective use of different redundancies such as hardware, time, and space redundancy.

In [20], a FMF based on neighbors cooperation, has been provided. Here, the nodes called gateways are distributed to monitor *CHs*. These nodes apply majority voting to diagnose faulty *CHs*. In recovery phase, on detecting a faulty *CH*, the gateway node selects itself as a new *CH*. In [21], the framework selects a *CH* and a cluster manager for each cluster. The cluster manager monitors the evaluation of the cluster. The cluster manager sends periodical messages to *CHs*, and when *CHs* do not respond, their fault is diagnosed. In recovery phase, on detecting a faulty *CH*, the manager selects a new *CH* from cluster members. A FMF based on the neighbors cooperation is represented in [22] and [23]. In these frameworks, the nodes have been cellularized, *CHs* diagnose faulty cluster members, and cell managers cooperate to manage faulty *CHs*. In recovery phase, the cell manager introduces itself as a new *CH* to the center and cluster members. In [24], fault management for *CHs* is discussed. In this method, dynamic and static backups are used for *CHs*. A reliability model based on the Markov model has been developed to evaluate clusters.

In [25], the Naive Bayes method is used for the fault tolerance of *CHs*. In this method, *CHs* evaluate the data received from its cluster members using the Naive Bayes method to detect faults. In [26], a method based on multi-hop paths for fault tolerance in heterogeneous networks is proposed. This method has two basic steps, the first step is spare routes for the main route and the second step is choosing the best route to send data. In [27], a fault diagnosis method based on an adaptive fuzzy neural inference system is proposed. In this method, the nodes with faults are classified using the adaptive fuzzy neural inference method. In [28], the authors use a blockchain-based algorithm for fault tolerance in nodes. This method uses the Pinocchio algorithm to evaluate node data. Therefore, by comparing each data with the neighboring data, the fault is detected. In [29], the authors used a machine learning-based method for fault tolerance. This method evaluates the data received from the nodes using a support vector machine. After evaluating the data, the node is determined to be normal or faulty.

3.1 Motivation

One of the most important challenges to FMFs in WSNs is increased energy consumption of nodes due to FM steps. On the other hand, reduced energy consumption increases the likelihood of occurring faults in nodes, which emphasizes the need for management. Thus there is a close relationship between FM and improved energy consumption. The general idea of the proposed framework is combining fault management and improving energy consumption methods in WSNs. In addition, the majority of fault management frameworks focus only on *CHs* faults, while each cluster member node may later be selected as a cluster head. Thus providing a fault management framework capable of detecting faulty *CHs* and cluster members is vital. On the other hand, on detecting a faulty node, most of the fault management frameworks remove it from network, which results in reduced number of active nodes. Thus it is essential to reuse faulty nodes in recovery phase.

4 Network, fault, and energy models in the proposed framework

In this section, network, fault, and energy models applied in the proposed method are discussed.

4.1 Network model

WSN is a specific wireless communication system, which does not rely on any fixed communication facilities. n nodes with a random uniform distribution density of size λ are distributed in a field of size $m \times m$. Nodes are homogeneous, and their positions are not predetermined [15]. Also, their radio radius is R_{max}. BS is located at a point far from the field. Nodes and BS are stationary. Each node applies various power levels to communicate with different nodes, and the operation time in the whole network can be divided into rounds. Since our framework is hierarchical- based, all nodes should be clustered, and CH_i is required for each cluster. Local synchronization of cluster members can be achieved by transferring a few bites in each cluster. Where $C_i = \{CH_i, cm_1, \ldots, cm_i\}$ denotes cluster member nodes and cluster heades.

4.2 Fault model

Based on the components, faults can occur at 3 levels: node, network, and BS [30]. The faults can be hardware or software destructions. Also, based on their duration, they can be classified into permanent, transient, and intermittent faults [31]. The faults of sensors resulting in general inactivity of the nodes are defined as permanent faults. Permanent faults are continuous and cannot be rectified. Sometimes the faults in the internal elements of sensors do not result in disconnection from other network nodes, but their data is incorrect, and the consequences can be transient or intermittent [32]. Transient faults are not permanent, and sometimes they are because of environmental changes. They occur in a so short time slot and are spontaneously rectified, but reoccur. It is so difficult to diagnose and manage transient faults. Intermittent faults occur in a longer time slot compared to transient ones [33]. They occur in intermittent time slots, which are typically specific. Detecting and managing these faults are easier. In the proposed framework, the faults that occur at network and nodes levels are detected and recovered. In fact, the faults of nodes such as battery depletion, transmitter and receiver circuit faults, and process and sensor unit's faults are managed. Also, the faults of network including faulty communication links are detected and recovered. Moreover, the faults classified based on their duration (i.e. permanent, intermittent and transient) are diagnosed through comparing the data of nodes.

4.3 Energy model

We adopted the radio model suggested in [34] to model the energy needed for sending and receiving data. In this model, the energy consumed by sensor i to transfer a message is derived from Eq. (1).

$$Energy_{ut} = \begin{cases} \left(\tau_t + \tau_{d_1}\rho^2\right) l_i;\rho < d_0 \\ \left(\tau_t + \tau_{d_2}\rho^4\right) l_i;\rho \geq d_0 \end{cases} \quad (1)$$

where $\tau_t\left[{}^{j}/_{\text{bit}}\right]$ denotes the energy bits consumed for transferring data. $d_{1\left[\frac{j}{bit}/m^2\right]}$ and $d_{2\left[\frac{j}{bit}/m^4\right]}$ denote the energy consumed by amplifier in an open space and multiple routing model [35]. $d_0 = \sqrt{\frac{\tau_{d_1}}{\tau_{d_2}}}$ of parameter ρ is the average of transferring distance from CH_i to the node and l_i denotes the message size, which is sent by each node. The energy consumed by sensor node i to receive a message is derived from Eq. (2).

$$Energy_{ur} = (\varepsilon_r L_i) \quad (2)$$

$\varepsilon r\left[{}^{j}/_{\text{bit}}\right]$ indicates the energy consumed by receiver circuit in each bit.

5 The steps of the proposed HFMF

In this section, the details of the steps of proposed algorithm, HFMF, to improve energy consumption and increase fault tolerance in WSNs is discussed.

5.1 Clustering and selecting *BCHs*

Hierarchical clustering algorithms are consisted of 2 phases: set up and steady state. In the proposed framework, when clusters are formed, the distance between nodes and CH is calculated. $\left(x_i, y_i\right)$ denotes the position of each node, and Euclidean distance can be used to derive the distance between CH_i and each cm_i.

On calculating the distance of nodes, cluster member nodes are ordered based on $min(dis_{i,j})$, and their list is stored in CH. Also, the residual energy of nodes is gathered by CH as well. Based on their minimum distance to CH and residual energy, a cluster member node is selected as the backup cluster head (BCH).

In steady state phase, when $BCHs$ are selected and data is aggregated by CH, a copy of data is sent to BCH. On receiving data by BS, it sends back an acknowledgement [24]. The copy is stored in BCH until acknowledgment is received. On aggregating data in CH, new data is compared with old data and in case of any difference, a copy is sent to BCH.

5.2 Generating virtual cells and diagnosing transient and intermittent faults

In the proposed method, sensors producing correlated data are identified and located in virtual cells. Based on the list stored in CH and the number of cluster members, some nodes are selected as representatives (RNs). Where n indicates the number of cluster members, $\frac{n-1}{2}$is the number of RNs. Then RNs calculate their distance to neighboring nodes and compare it with radio range to define the overlap of radio range. If the distance between the RNs and their neighboring nodes is less or equal to 30% of radio range, their data will be evaluated in the next step. To derive data correlation, it is required to calculate the data correlation between nodes and their neighbors. Where $N(i)$ indicates the number of the neighbors of node i

whose radio range overlap, and j denotes a neighbor, their data difference is derived from $data_{ij}$ in Eq. (3).

$$data_{ij} = \sqrt{\left|D_i(t_1) - D_j(t_1)\right|^2 + \left|D_i(t_2) - D_j(t_2)\right|^2 + ..} \quad (3)$$

D_i denotes the data received by node i, and t_i indicates time. On calculating $data_{ij}$ in several time slots, we can derive expected value (E) from Eq. (4).

$$E(data_{ij}) = \frac{\sum data_{ij}}{N(i)} \quad (4)$$

Utilizing E, we can derive the standard deviation of data. Also, standard deviation indicates the nodes producing close data (Eq. 5).

The data is evaluated by RNs in defined slot time. If the nodes are not faulty, and their data is similar to that of cell RNs, they will be considered as cell member nodes. However, healthy nodes that their produced data is different from that of RNs can introduce themselves as an independent RN to CH. Provided that the data is in a suspicious range, it will be flagged in the next rounds to be evaluated and defined as a faulty or healthy node. When there is not a data correlation between nodes and their neighbors in different time slots, they are identified as faulty nodes. On diagnosing faults, CH is informed to ignore the data received from these nodes. Applying standard deviation, it is possible to diagnose stuck at one and zero faults. Actually, if the data of a node is lower or higher than the standard deviation of a neighboring node in several successive rounds, its fault will be diagnosed.

$$\sigma = E\left(data_{ij} - E(data_{ij})\right)^2 = E\left(data_{ij}^2\right) - \left[E(data_{ij})\right]^2 = {}^1/_{N(i)} \sum_{j \in N(i)} data_{ij}^2 - \left(\frac{1}{N(i)} \sum_{j \in N(i)} data_{ij}\right)^2 \quad (5)$$

By calculating this parameter, the nodes are identified as normal, suspect, and faulty. Also, it is possible to diagnose transient and intermittent faults in nodes. In the proposed method, on calculating E and σ of nodes, the data of them is placed in normal, suspect, and faulty time slots (Eq. 6).

$$D_i = \begin{cases} [E-\sigma, E+\sigma], normal \\ [E-2\sigma, E-\sigma] \cup [E+\sigma, E+2\sigma], suspect \\ ]-\infty, E-2\sigma[\cup]E+2\sigma, +\infty[, faulty \end{cases} \quad (6)$$

Cluster member nodes generating similar data to that of RNs are located in virtual cells; other cell members including spare, sleep, and RN nodes are active to monitor area. *CHs*,*BCHs* and *RNs* are active; cluster member nodes whose data is correlated with RNs are as *BRNs* in sleep nodes. Thus sleep mode is applied on demand. The schematic outline of cluster members is illustrated in Fig. 1. Here, faulty nodes are identified and considered as dead nodes. When nodes are in sleep mode, energy consumption is improved and occurring faults is reduced. Also, reduced number of active cluster member nodes makes FM easier.

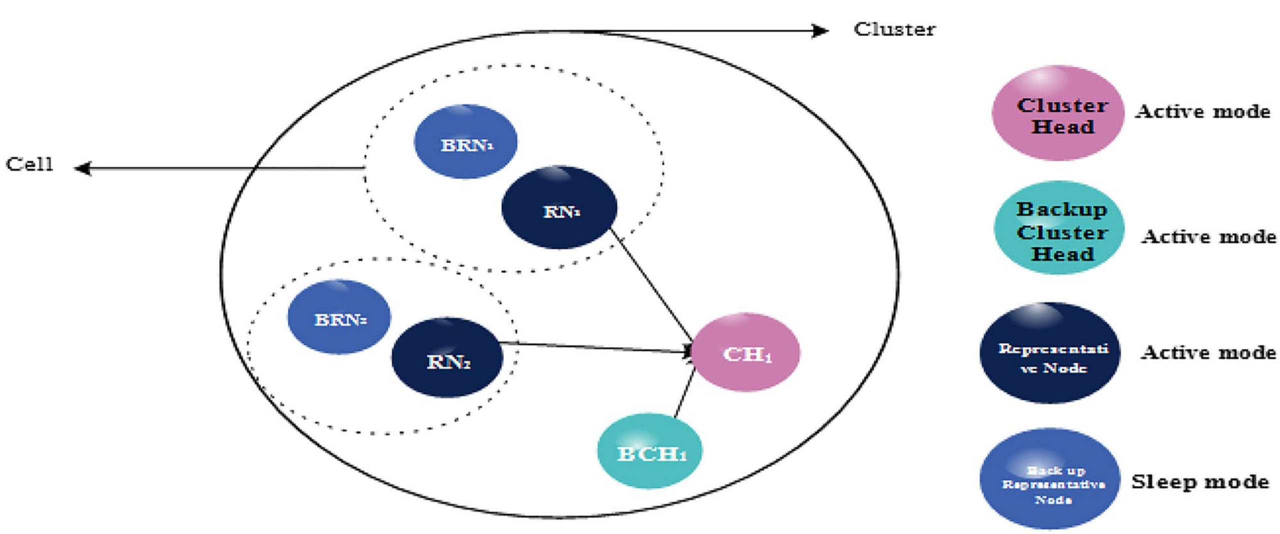


Fig. 1 Formation of cells in the proposed framework

5.3 Fault detection and recovery in HFMF

In this phase, FM and self-detection are employed to diagnose permanent faults with minimum delay. Due to the probability of occurring faults in the battery, sensor unit, process unit, transmitter circuit, and receiver circuit of nodes, occurring faults in these components is evaluated.

Battery fault detecting: First, occurring faults in the battery of nodes is evaluated. *RNs* can detect the fault of battery based on their own residual energy. By calculating the energy consumed to send and receive messages, the residual energy is derived. When the residual energy is less than a threshold, RNs send a message to their own *CH* to inform it, and the fault is diagnosed.

Process and sensor fault detecting: Nodes are consisted of a set of sensor units, which send sensed data to process unit. To guarantee a proper performance, the data of all nodes should be evaluated. Here, hypothesis testing is used to evaluate the condition of individual nodes. Here, the temporal correlation between nodes is applied to detect faults in the process unit or the sensor unit of nodes. Where the data sensed by i^{th} node in time t is indicated by D_i^t, and U_i^t denotes the binary decision of i^{th} node, the data of nodes is conditional and can be derived from Eq. (7).

$$P\left(D_1, D_2, D_3, \ldots ..D_N | H_k\right) = \prod_{i=1}^{N} P(D_i | H_K), K = 0,1 \tag{7}$$

According to hypothesis testing, the same local decision-making rule is applied in all nodes. On gathering received data from environment, each *RN* makes a binary decision based on local decision-making rule and sends it to its own *CH*. Decision U of i^{th} node is calculated through Eq. (8) and local decision-making rule (δ). When the decision is in favor of H_0, decision “0” is sent, otherwise decision “1” is sent.

$$U_i^t = \delta(D_i^t) \tag{8}$$

The decisions of *RNs* are listed in a record table and then the rate of transferred binary decisions of nodes is derived from Eq. (9).

$$A_i^t = \frac{1}{t} \sum_{K=1}^{t} U_i^K \tag{9}$$

CHs apply majority voting in specific time slots to detect faulty nodes. In fact, the rate of received binary decisions of each node is compared with that of others. Thus the faults in process unit and sensor unit can be diagnosed.

Transmitter and receiver circuit fault detecting: The faults are diagnosed by *CHs*. In specific time slots, each *RN* sends a heartbeat message (HB) of 200 bits to *CH*, and *CH* sends back an ACK of the same size. Then *CH* evaluates the condition of the transmitter circuit of node i in time S_t.

$$Tc_i = \frac{\sum (Number\ of\ heartbeat\ messages\ sent\ to\ CHs)}{timeslot} = \frac{\sum |Hb|}{S_t} \tag{10}$$

Then *CH* makes a comparison between the result and a threshold. When the value is less than the threshold, the fault of the transmitter circuit of node is diagnosed.

Due to the necessity of receiving ACK for each message, sent by *CH*, the receiver circuit can be evaluated by the node itself. The condition of receiver circuit is derived from Eq. (11). In fact, the number of received ACKs in a specific time slot is calculated. If the result is less than a threshold, the fault of receiver circuit will be diagnosed, and *CH* will be informed by a message.

$$Rc_i = \frac{\sum (Number\ of\ Ack\ messages\ sent\ by\ CHs)}{time\ slot} = \frac{\sum |Ack|}{S_t} \tag{11}$$

Fault recovery in RNs: On diagnosing the faults of *RNs*, the recovery phase initiates. Here, *CH* selects one of the closest nodes among spare cell members which are in sleep mode. Next, by receiving a message, the node turns to active mode and replaces the faulty node as a new *RN*.

Detecting and recovering faults in CHs: Since *CHs* play an important role in aggregating and sending data to BS, detecting and recovering their faults are necessary. In the proposed framework, there are 2 methods to detect the faults of *CHs*. In the first method, *BCH* evaluates the battery, the transmitter circuit, and the receiver circuit of *CH* periodically. Thus any faults in these components is immediately detected by *BCH*. In the second method, when BS does not receive any data from a *CH*, the *CH* is detected faulty. In fact, if *BCH* is faulty and cannot diagnose the faults of *CH*, BS will detect the faults.

On detecting the faults of *CHs*, recovery phase initiates. In the first method, when *BCH* detects a fault in a *CH*, it informs BS through sending a message. Next, *BCH* replaces the faulty *CH* and then announces the change to other cluster members so that they send their data to *BCH*. Also, based on the energy and distance a cluster member is selected as a new spare. If the spare node is a cell member which is in sleep mode, it will be turned to active mode through sending a message. The new spare node monitors *BCH* and stores a copy of data. In the second method of detecting faults, when BS detects a faulty *CH*, it checks the condition of *BCH*. Provided that *BCH* is healthy, it will replace CH. Then other cluster members will be informed, and they select their own BCH. However, if the new BCH dose not send a message to BS, reclustering will be performed.

Detecting and recovering faults in communication links: The proposed FD allows diagnosing faults in communication links between CH and RNs, CH and BS, and CH and spare node. The faults of the communication links between nodes can be diagnosed through propagation speed, reliability of link, and lack of receiving ACKs. $V(l_i)$ denotes the propagation velocity of a link. Propagation speed is the period of time that a bit needs to travel from the beginning point to the end point (Eq. 12).

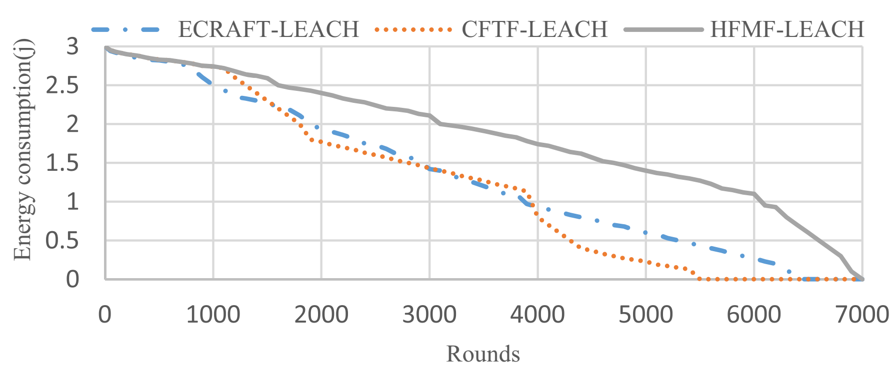
$$V(l_i) = \frac{dis_{i,j}}{\left[\left(D_{MAC} + D_{queue} + D_{Transmission}\right) * C_{ij}\right]} \tag{12}$$

where $dis_{i,j}$ denotes the distance between 2 nodes, and D_{TX}, D_{queue}, and D_{MAC} indicate the delay of transfer, queue, and MAC accessibility channel, respectively. Moreover, $C_{i\,j}$ indicates the transmission counts between node i and j. Regarding this parameter, when propagation speed is less than a threshold, the fault of communication link is diagnosed.

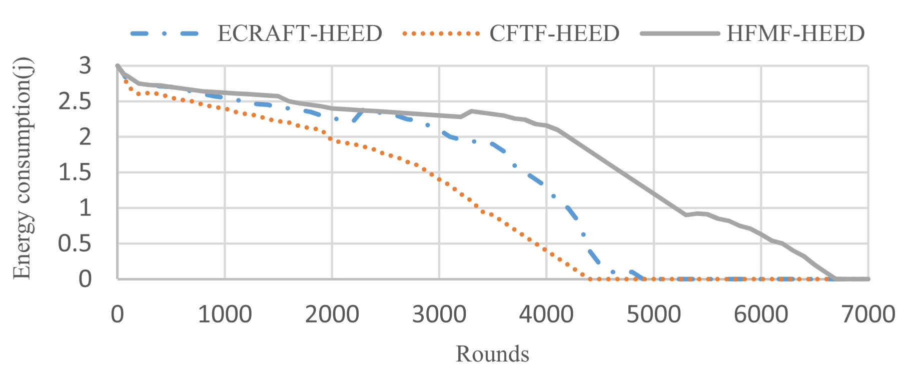
Table 1 Simulation parameters

Value	Parameters
2000	Nodes number
2000*2000	Network size
1000*1000	BS Area
100 m	Communication range
2000 bites	Data packet
3.0j	Initial energy of sensor nodes
12 (mW)	Idle power
13(W)	Sleep power
50 nJ/bit	E_{elec}
10 pJ/bit/m^2	ε_{fs}
87.0 m	d_0

Increased reliability of links results in reduced faults. However, when the reliability of a link is less than a threshold, faults can occur in it. In addition to propagation speed and reliability, when node does not receive ACKs, it can be an indication of faults in communication links between nodes. Regarding the communication link between CH and RN, when CH receives the message of the condition of transmitter and receiver circuits from RN and do not send back an ACK, a communication link fault will be diagnosed.



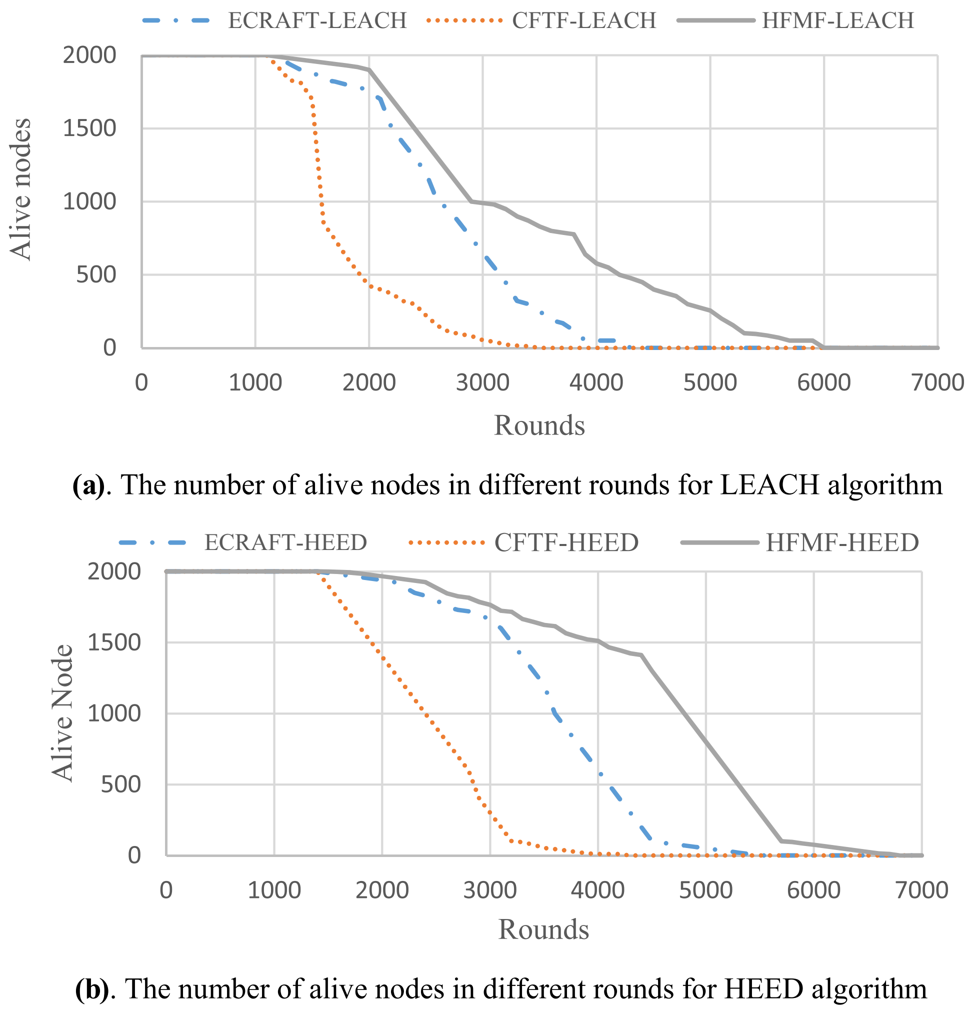
(a). The energy consumption of nodes in different rounds for LEACH algorithm



(b). The energy consumption of nodes in different rounds for HEED algorithm

Fig. 2 (**a**). The energy consumption of nodes in different rounds for LEACH algorithm (**b**). The energy consumption of nodes in different rounds for HEED algorithm

Fig. 3 (**a**). The number of alive nodes in different rounds for LEACH algorithm (**b**). The number of alive nodes in different rounds for HEED algorithm



Also, the communication link between *CH* and BS will be diagnosed faulty if BS receives the data of *CH* but does not send back an ACK or request for resending. To recover the communication links between nodes a spare node is selected to replace the faulty communication links.

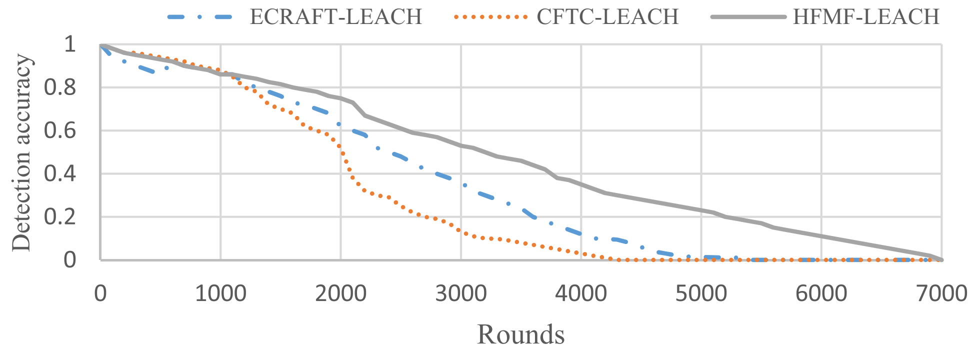
6 Results and evaluation

In this section, the performance of the proposed framework is evaluated through simulation. To evaluate the FMF, we compare it with 2 other FMFs: comprehensive fault-tolerant framework (CFTF) [19] and ECRAFT [36]. We applied MATLAB to simulate the framework. Simulation parameters have been listed in Table 1. For simulation, various parameters including consumed energy, the number of alive nodes, and FD accuracy are evaluated. To make a comparison, LEACH [37] and HEED [1, 38] algorithms are placed in these frameworks to evaluate their parameters. Both of these algorithms are clustering techniques, which have attracted the attention of authors. The energy consumption of the proposed method is discussed in Sect. 3.3. The ratio of the current network nodes to the total number of preliminary nodes is the number of alive nodes. The ratio of the number of correctly identified faulty nodes to the total number of actual faulty nodes is FD accuracy parameter. From round 0 to 1000, sensor fault probability equals 0.1, but it rises to 0.2 from round 1000 to 2000; this procedure continues to round 7000, and sensor fault probability reaches 0.7.

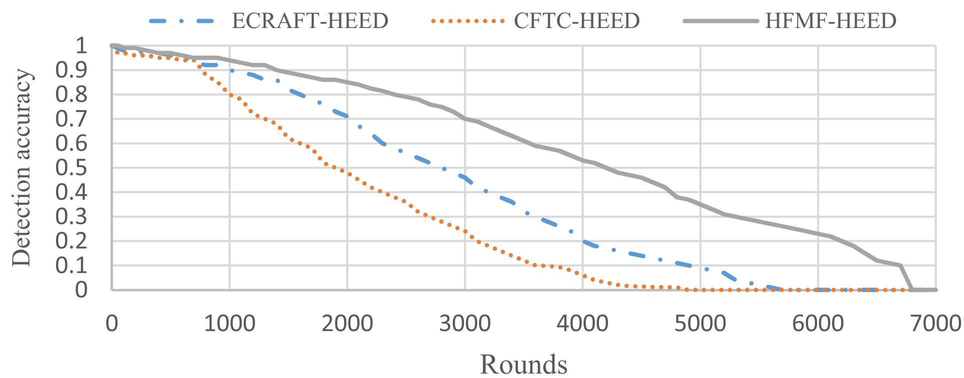
6.1 Experimental results

Figure 2(a) illustrates the energy consumption level of network nodes in different rounds for LEACH. In initial rounds, the proposed framework consumes the same level of energy as others to generate cells and place nodes in sleep/awake modes. However, in upper rounds, the proposed framework consumes less energy compared with others due to applying data correlation and sleep/awake methods. In ECRAFT, the waiting time needed for receiving each ACKs results in maintaining nodes, especially *CHs*, in active mode for a longer period of time. However, in CFTF, due to the

Fig. 4 (**a**). FD accuracy in different rounds for LEACH algorithm (**b**). FD accuracy in different rounds for HEED algorithm



(a). FD accuracy in different rounds for LEACH algorithm



(b). FD accuracy in different rounds for HEED algorithm

necessity to send successive synchronization messages and the messages of evaluating the condition of nodes, the level of energy consumption is higher. Figure 2(b) illustrates the energy consumed in different rounds for HEED algorithm. In ECRAFT and CFTF frameworks, the level of energy consumption is higher. In fact, ECRAFT needs frequent updating, which causes *CH* and *BCH* to consume more energy. However, the proposed framework needs updating when the data in *CH* is changed. In CFTF, there is not any mechanism to improve energy consumption and in case of a faults in *CH*, reclustering is performed. Thus applying FD methods leads to increased energy consumption of nodes.

Figure 3(a) and (b) illustrate the number of alive nodes in different rounds. In LEACH, when we apply these 3 aforementioned frameworks, the number of residual nodes from round 0 to 1000 equals the total number of nodes. However, there is an increase in the number of nodes in upper rounds. In the proposed framework, self-detection allows nodes to evaluate their own condition. In different algorithms, there is a direct relationship between energy consumption and the number of nodes in rounds. Since FD frameworks cause a rise in energy consumption, maintaining a balance between energy consumption and FD results in a rise in the number of residual nodes. As shown in Fig. 3(a) and 3(b), in HFMF, the number of residual nodes in different rounds is more than ECRAFT and CFTF since the energy consumption of HFMF is decreased. The proposed framework outperforms other frameworks in rounds above 3000. Since ECRAFT adopts simple methods to detect and recover faults, the energy consumption is reduced; however CFTF sends frequent messages to detect faults, which leads to increased energy consumption and decreased number of residual nodes. In the proposed framework, the number of residual nodes for HEED algorithm is more than other frameworks, which is due to relying on residual energy for adopting *CHs* and *BCHs* and correlating cells to reduce sending repeated data to *CH* (Fig. 3(b)).

FD accuracy of LEACH algorithm in different rounds for these 3 frameworks has been shown in Fig. 4(a). The accuracy of FD in the proposed framework is superior to others since we have combined self-management and *CH* management methods. Moreover, decreased number of active nodes leads to declining the likelihood of occurring faults in sleep nodes. Also, as shown in Fig. 4(b), the accuracy of FD in the proposed framework is higher compared with ECRAFT and CFTF. The main difference between the proposed framework, ECRAFT, and CFTF refers to its high level of accuracy, especially in upper rounds, where the probability of occurring faults is high. In the initial steps, the proposed framework can diagnose transient faults through data correlation between neighboring nods. Then nodes report their condition to their *CH* periodically so that their permanent faults are detected. This process leads to a higher level of FD accuracy. In ECRAFT, the faults related to the battery depletion of nodes are detectable. Also, in CFTC, the same faults or permanent faults preventing message transferring are detectable. In ECRAFT and CFTC, the accuracy of FD in rounds over 5000 is almost 0. However, in the proposed framework, the accuracy of FD in rounds over 6000 is approximately 0.

6.2 limitations of the research

Although the energy consumption is expected to increase in fault management frameworks, in the proposed method the energy consumption was improved with the help of the sleep/active method. Therefore, the most important advantage of the proposed method was the improvement of energy consumption and increase of active nodes compared to other frameworks. On the other hand, FD accuracy was increased with the help of self-detection method and increasing the cluster hierarchy. One of the limitations of the proposed framework is that this method can only be implemented in hierarchical clustering algorithms and is not suitable for networks with a low number of nodes. On the other hand, the faults that occurred in the base station and the sink cannot be identified by the proposed method because our focus was on the faults that occurred in the nodes and links Therefore, in the future we will try to provide a framework to overcome these limitations.

7 Conclusion

A hierarchical fault management framework (HFMF) has been proposed in this paper. The proposed fault management framework seeks to detect and recover faults through combining the methods of improving energy consumption and fault management, and also minimizes the energy consumption of nodes. First, nodes are clustered and cellularized. A *CH* and a *BCH* are selected for each cluster. *BCH* monitors the performance of cluster and replaces the faulty *CH*. Here, all permanent and intermittent faults of nodes are detectable and recoverable by the node itself and *CH*. The results of simulation reveal that the proposed framework outperforms ECRAFT and CFTC in terms of energy consumption, number of alive nodes, and FD accuracy. Applying sleep-awake method in the proposed framework resulted in improved energy consumption, reduced probability of occurring faults in nodes, and increased number of alive nodes in different rounds. Also, due to producing several methods to detect permanents and intermittent faults in networks and nodes, the accuracy of the framework is superior to others.

References

1. Moridi E et al (2020) Fault management frameworks in wireless sensor networks: A survey. Comput Commun
2. Patil K, De Turck K, Fiems D (2019) Optimal data collection in wireless sensor networks with correlated energy harvesting. Ann Telecommun 74(5–6):299–310
3. Muhammed T et al (2020) HCDSR: A hierarchical clustered fault tolerant routing technique for IoT-based smart societies. Smart Infrastructure and Applications. Springer, pp 609–628
4. Lin J-W et al (2019) Efficient Fault-Tolerant Routing in IoT Wireless Sensor Networks Based on Bipartite-Flow Graph Modeling. IEEE Access 7:14022–14034
5. Javaid A et al (2019) Machine Learning Algorithms and Fault Detection for Improved Belief Function Based Decision Fusion in Wireless Sensor Networks. Sensors 19(6):1334
6. Guo Y, Lv R, Li Z (2020) A realized on-demand cross-layer connection strategy for wireless self-organization link based on WLAN. Peer-to-Peer Networking and Applications 1–15
7. Fu X et al (2021) Lightweight Fault Detection Strategy for Wireless Sensor Networks Based on Trend Correlation. IEEE Access 9:9073–9083
8. Moridi E et al (2020) Novel fault-tolerant clustering-based multipath algorithm (FTCM) for wireless sensor networks. Telecommun Syst 74(4):411–424
9. Abdullah-Al-Wadud M, Hamid MA (2014) A fault-tolerant structural health monitoring protocol using wireless sensor networks. annals of telecommunications-annales des télécommunications 69(3–4):219–228
10. Yadav SA, Poongodi T (2021) A Review of ML Based Fault Detection Algorithms in WSNs. in 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM). IEEE
11. Moussa N, Hamidi-Alaoui Z, El Alaoui AEB (2019) CFTM: A centralized fault tolerant mechanism for wireless sensor networks. in 2019 5th International Conference on Optimization and Applications (ICOA). IEEE
12. Jan H et al (2015) Dependability and reliability analysis of intra cluster routing technique. Peer-to-Peer Networking and Applications 8(5):838–850
13. Shankar A et al (2020) Increasing fault tolerance ability and network lifetime with clustered pollination in wireless sensor networks. J Ambient Intell Humaniz Comput 1–14
14. Saleh I, El-Sayed H, Eltoweissy M (2006) A fault tolerance management framework for wireless sensor networks. in 2006 Innovations in Information Technology. IEEE
15. Baskar S, Dhulipala V (2018) M-CRAFT-modified multiplier algorithm to reduce overhead in fault tolerance algorithm in wireless sensor networks. J Comput Theor Nanosci 15(4):1395–1401
16. Yin, M.H., Win ZJIJoFC (2014) and Communication, Fault management using cluster-based protocol in wireless sensor networks. 3(1):36
17. Hamdan D et al (2012) Integrated fault tolerance framework for wireless sensor networks. in 2012 19th International Conference on Telecommunications (ICT). IEEE
18. Alam MM, Mamun-Or-Rashid M, Hong CS (2008) Wsnmp: A network management protocol for wireless sensor networks. in 2008 10th International Conference on Advanced Communication Technology. IEEE
19. Afsar MJS, Networks C (2015) A comprehensive fault-tolerant framework for wireless sensor networks 8(17):3247–3261
20. Syed M, Dubey M (2020) Compendious elucidation on faults management in wireless sensor networks (wsn). in 2020 International

Conference on Computation, Automation and Knowledge Management (ICCAKM). IEEE
21. Bagheri T (2012) DFMC: decentralized fault management mechanism for cluster based wireless sensor networks. in 2012 Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP). IEEE
22. Babaie S, Rasi TJIJoCS (2011) and I. Security, DCMC: Decentralized and Cellular Mechanism for improving fault management in Clustered wireless sensor networks. 9(11):158
23. Asim M, Mokhtar H, Merabti M (2009) A cellular approach to fault detection and recovery in wireless sensor networks. in 2009 Third International Conference on Sensor Technologies and Applications. IEEE
24. Tong Y et al (2020) Fault Tolerance Mechanism Combining Static Backup and Dynamic Timing Monitoring for Cluster Heads. IEEE Access 8:43277–43288
25. Chu S-C, Dao T-K, Pan J-S (2020) Identifying correctness data scheme for aggregating data in cluster heads of wireless sensor network based on naive Bayes classification. EURASIP J Wirel Commun Netw 2020(1):1–15
26. Deniz F, Bagci H, Korpeoglu I (2021) Energy-efficient and fault-tolerant drone-BS placement in heterogeneous wireless sensor networks. Wireless Netw 27(1):825–838
27. Raja MS et al (2021) Diagnosis of fault node in wireless sensor networks using adaptive neuro-fuzzy inference system. Appl Nanosci 1–9
28. Zeng C et al (2020) Pinocchio: A blockchain-based algorithm for sensor fault tolerance in low trust environment. in 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE
29. Jan SU, Lee YD, Koo IS (2021) A distributed sensor-fault detection and diagnosis framework using machine learning. Inf Sci 547:777–796
30. Kirubakaran J et al (2020) Delay sensitive aware distributed data fault recognition algorithm for distributed sensor networks. Peer-to-Peer Networking and Applications 13(4):1080–1090
31. Tang Y, Cheng G, Xu Z (2011) Probabilistic and reactive fault diagnosis for dynamic overlay networks. Peer-to-Peer Networking and Applications 4(4):439–452
32. Gobinath T, Tamilarasi A (2020) RFDCAR: Robust failure node detection and dynamic congestion aware routing with network coding technique for wireless sensor network. Peer-to-Peer Networking and Applications 13(6):2053–2064
33. Menaria VK et al (2020) NLFFT: A novel fault tolerance model using artificial intelligence to improve performance in wireless sensor networks. IEEE Access 8:149231–149254
34. Heinzelman WB, Chandrakasan AP, Balakrishnan HJITowc (2002) An application-specific protocol architecture for wireless microsensor networks. 1(4):660–670
35. Mosavifard A, Barati H (2020) An energy-aware clustering and two-level routing method in wireless sensor networks. Computing
36. Cheraghlou MN, Khadem-Zadeh A, Haghparast MJWPC (2017) Increasing Lifetime and Fault Tolerance Capability in Wireless Sensor Networks by Providing a Novel Management Framework. 92(2):603–622
37. Heinzelman WR, Chandrakasan A, Balakrishnan H (2000) Energy-efficient communication protocol for wireless microsensor networks. in Proceedings of the 33rd annual Hawaii international conference on system sciences. IEEE
38. Younis O, Fahmy SJITomc (2004) HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. (4):366–379

Elham Moridi received B.Sc degree in Information technology (IT) Engineering from Dezful Branch, Payame Noor Universtiy, Dezful,Iran, in 2011. She received her M.Sc. degree in Computer Systems Architecture Engineering from Islamic Azad University, Dezful Branch, Iran in 2015. She received her Ph.D. degree in Computer Systems Architecture Engineering from Islamic Azad University, Science and Research Branch, Iran in 2020. Her Research interests include Wireless Sensor Networks, Mobile Ad-Hoc networks and Vehicle Ad-hoc networks.

Majid Haghparast received his B.Sc. in computer hardware engineering in 2003. He received his M.Sc. and Ph.D. degrees in computer architecture in 2006 and 2009, respectively. Since 2007, he has been affiliated with the Computer Engineering Faculty, Yadegar-e-Imam Khomeini (RAH) Shahre Rey Branch, IAU University, Tehran, Iran. He is currently an associate professor and the head of the computer engineering department at Electrical and Computer Engineering Faculty, Yadegar-e-Imam Khomeini (RAH) Shahre Rey Branch, IAU University, Tehran, Iran. He has also been selected as his university's best researcher in 2017. Majid has published more than 50 research papers in various international journals and conferences. His research interests include cloud computing, WSNs, reversible logic, and computer arithmetic. Since April 2017 he is conducting his sabbatical at the Johannes Kepler University Linz, Austria, where he also is a Research Fellow. He served in more than 40 international conference advisory and technical program committees. Dr. Haghparast is on the panel of reviewers for various international journals.

Mehdi Hosseinzadeh received his B.S. degree in computer hardware engineering, from Islamic Azad University, Dezfol branch, Iran in 2003. He also received his M.Sc. and the Ph.D. degree in computer system architecture from the Science and Research Branch, Islamic Azad University, Tehran, Iran in 2005 and 2008, respectively. He is currently an Associate professor in Iran University of Medical Sciences (IUMS), Tehran, Iran, and his research interests include SDN, Information Technology, Data Mining, Big data analytics, E-Commerce, E-Marketing, and Social Networks.

Somayyeh Jafarali Jassbi was born in Tehran, Iran, in 1983. He received the B.E. degree in computer architecture engineering in 2007, and the Ph.D. degrees computer architecture engineering in 2010 from the Azad University Science and Research branch, Tehran, Iran.In 2010, she joined the Department of computer Engineering, Azad University science and research branch, she became an Associate Professor in 2011. Her current research interests include Cloud computing, Internet of Thing (IOT), Wireless sensor network and computer architecture. She was head of computer department in 2012. She write, translate and published several professional books and papers in her fields.