Wireless Netw (2017) 23:2263–2280
DOI 10.1007/s11276-016-1291-2

ZBFR: zone based failure recovery in WSNs by utilizing mobility and coverage overlapping

Krishna Pal Sharma[1] · **Teek Parval Sharma**[1]

Published online: 13 May 2016

Abstract Wireless sensor networks are more prone to failures as compared to other traditional networks. The frequent faults and failures sometime create large holes causing loss of sensing and connectivity coverage in the network. In present work, a zone based failure detection and recovery scheme is presented to reliably handle such node failures. We first propose a consensus and agreement based approach to elect a suitable monitor node called as zone monitor (ZM). ZM is responsible for coordinating failure recovery activities and maintaining desired coverage within a zone. In order to overcome failure overhead due to false failure detection, a consensus is carried out amongst neighboring nodes of a suspicious node to confirm the correct status with high accuracy. On confirmation of a node failure, the impact of resulting hole on coverage is analyzed and if impact exceeds beyond a particular threshold, a recovery process is initiated. The recovery process utilizes backup nodes having overlapping sensing coverage with failed node and may also relocate some nodes. Firstly a backup node is probed and activated if available. If no backup node is found, the solution strives to recover coverage jointly by recursively relocating some mobile nodes and probing backup nodes. The proposed scheme is analyzed and validated through NS-2 based simulation experiments.

Keywords Wireless sensor network · Coverage · Connectivity · Failure diagnosis · Failure recovery

✉ Krishna Pal Sharma
kpsharma17vce@gmail.com

[1] Department of Computer Science and Engineering, National Institute of Technology, Hamirpur, HP, India

1 Introduction

The advancements in various technologies like wireless communication, system-on-chip (SoC), and Micro-Electro-Mechanical Systems (MEMS) have facilitated the development of intelligent sensors (e.g. Tmote Sky from Mote IV, Mica motes from Crossbow, the MKII nodes from UCLA, etc.). Also, the concept of mobile sensors has been spurred by the recent advancements in distributed computing and robotics technology. In recent times, the availability of such versatile range of sensor nodes has resulted in diverse applications of wireless sensor networks (WSNs) and has motivated lots of researchers in this field [1, 2]. For WSN applications such as coastal and border protection, battlefield surveillance, combat field reconnaissance, environmental monitoring, it is envisioned that a set of mobile and static sensor nodes are randomly deployed [3, 4]. For such applications, sensor nodes are generally deployed in hostile and remote locations, where failures occur more frequently and unexpectedly as compared to other traditional networks. Failures may occur due to various reasons including battery exhaustion, radio interference, de-synchronization, or dislocation [5, 6]. If not handled timely, these failures may cause connectivity loss leading further to network partitioning. In case of clustered networks, node failures create large holes in the network, causing data and connectivity loss [7].

In WSNs, connectivity and coverage of the network are crucial throughout the complete network operation time for meeting the desired application requirements. For instance, if WSNs need to periodically transmit the sensed data successfully to the sink node, then the network should be connected at all times. Likewise, for monitoring all spots of a region accurately, the network should provide desirable coverage during entire lifetime of the network. In presence of frequent failures, providing desired quality of service (QoS) is more challenging in remotely deployed WSNs, where manual replacement of failed node is not possible. Therefore, in order to provide self-healing capability against such failures, mobile sensor nodes are extensively considered as an option. Although, the use of mobile nodes in sensor networks increases the cost of the network, but is very useful for connectivity restoration, replacing failed nodes and dynamic adaptation of the network. Mobile nodes are much more versatile than static sensor nodes as they can be deployed in any scenario and can cope with rapid topological changes. Instead, a combination of mobile and static sensor nodes has generally been preferred. Also, the use of mobile sensor nodes is quite evident from some recent schemes proposed for handling failures and topology management where such nodes are extensively used [8–10].

In recent past, many approaches have been proposed for handling and recovering from node failures. But most of them focus on connectivity coverage rather than sensing coverage issue whereas sensing coverage is a major attribute towards QoS. Also, for every node failure in the network many schemes exploit node mobility to provide node replacement for failed node [9–11]. Such node movements consume lot of energy and hence must be minimized. Generally, the existing failure handling approaches detect failures through listening of heartbeat massages. If the heartbeat message from a particular sensor node is not heard within a time bound then the node is considered failed. While in case of unreliable wireless channel as used in WSNs, the heartbeat messages may be lost due to many reasons like, congestion, collision, hindrance, etc. and can cause wrong failure identification. The unnecessary movements due to such false detections can be avoided by enhancing the failure detection accuracy. Even after the movement, network has to suffer from topological changes which may also be costlier in terms of energy consumed in updating routing information. Therefore, a very low false alarm rate mechanism for detecting failure is desirable for such networks.

In case of random deployment, WSNs are densely deployed, where many nodes overlap the coverage with their neighbors. In such WSNs, if the coverage area of a sensor node is fully covered by its neighbor, then the sensor node can act as backup node (BN) [12]. Many scheduling algorithms [12, 13] utilize such backup nodes for lifetime enhancement of the network through implementing turning on/off schedule of nodes. Therefore, in some cases the failure of a single or multiple nodes may be handled by just changing the state of backup nodes from turn-off to turn-on i.e. mobility is needed only when there is no schedule available to handle the failure. Hence, unnecessary movements of nodes can also be avoided by utilizing overlapping coverage.

For large networks, centralized handling of failures is not desirable due to creation of single point of failure and generation of high volume of information/communication near central node [14]. Whereas, fully distributed approaches have lots of communication overhead for coordinating tasks and collecting global information about coverage and connectivity of the network [15]. Therefore, we consider a network divided into several sub-regions called zones or clusters and failures in every zone is handled individually by a zone monitor (ZM). The ZM is responsible for detecting failures accurately and recovering from them timely so that the coverage above a threshold and at least 1-connectivity can be maintained throughout the lifetime of the network.

The proposed failure handling approach has mainly three phases: election of ZM, failure detection, and failure recovery. Firstly, a priority based ZM election mechanism is proposed, where priority is a function of distance of ZM from zone centroid and remaining energy. The scheme always elects an energy rich and communication efficient node as ZM amongst all available sensor nodes. For balanced energy dissipation amongst all nodes, election is performed either when ZM's energy reaches at a threshold or when it fails. Secondly, a majority voting based failure detection strategy is proposed. The scheme uses heartbeat message for failure detection and marks a node as suspicious if a heartbeat message is not received from it. Once a node is marked suspicious, its confirmation is done through a voting procedure performed amongst neighbors of that suspicious node. Finally, an energy efficient failure handling mechanism is presented which first looks for the backup node and if not available, exploits cascaded movement of mobile nodes.

The rest of the paper is organized as follows. In the next section, a brief summary of the related work is given. Section III elaborates on system model and major assumptions. In section IV, the proposed scheme is described. The section V gives performance evaluation followed by conclusion in section VI.

2 Related work

Mainly two types of schemes are available in literature: proactive and reactive [10]. The proactive schemes assign resources in advance in the network such that failures can be tolerated. This augmentation is mainly done at the time of setup or during normal operation of the network. Two variants of such approaches are available. At first, the network topology is designed in such a way that the network can tolerate failures without degrading desired coverage and connectivity [16–19]. In the second variant, some sensor nodes are strategically augmented in network topology for tolerating failures [11, 20]. While in case of reactive schemes, to cope with a failure the solution strives dynamically after its occurrence. Further in this category, three variants exist. In first category, mobile nodes are used to recover the network from failures or network partitioning [8–10]. In the second category, some sensor nodes are re-deployed strategically in the network for recovering from connectivity and coverage losses. While in third category, some mobile relay nodes are deployed in the network, which tour disjoint block of nodes and carries data between them [20, 21].

2.1 Proactive schemes

The objective of popular proactive strategies is to form k-connectivity in the network for $k \geq 2$. The redundant nodes are placed for creating more than one path between each pair of sensor nodes for mitigating failures. Such schemes rely on tolerating failures rather than recovering them. The formation of such topology is very challenging and has been proven NP-hard even for $k = 1$ [16]. The complexity of such schemes is tackled through many sub-optimal heuristics [16, 17]. Lin et al. [16] propose sub-optimal solution by using graph theory. The network is considered as a graph, where each edge is assigned a weight which represents the number of nodes required to be placed for establishing the required k-connectivity. An improved version of the approach of [16] is proposed by Li et al. in [19]. In schemes proposed by Vaidya et al. [20] and Akkaya et al. [8], the redundant nodes are assigned to each critical node say cut-vertex node as backup nodes. Bagci et al. [22], present a k-connected topology formation mechanism for heterogeneous WNS having several super nodes with unlimited energy resources. The approach aims to form k-connectivity amongst super nodes and other sensor nodes by assigning appropriate communication range to each node. Huang et al. [23] propose a novel Fuzzy-logic Topology Control (FTC) approach for constructing k-connectivity by adjusting communication ranges of nodes. In the scheme, the decision of communication range adjustment is based on dynamically generated training data set. Results reveal that the scheme is able to construct desired node connectivity. The scheme assigns comparatively low communication range to most communicating nodes in order to balance energy drain in the network. Most of schemes discussed in this section form strong connectivity at the time of network deployment. In WSNs, failures are very frequent and un-deterministic, hence such types of schemes generally are not considered suitable.

2.2 Reactive schemes

Reactive schemes are most suitable for dynamically changing topology and for un-deterministic failure scenarios [8–10]. Abbasi et al. [8], propose a distributed actor recovery algorithm (DARA). Whenever a cut-vertex node fails, DARA relocates failed node's neighboring node to failed node's location in order to reconnect the network. This relocation of neighboring node may further cause network partitioning and hence, the relocation process is applied recursively until a leaf node is encountered. The scheme finds solution locally by searching 1-hop and 2-hop neighbor information of the failed node. In order to minimize overall node displacements due to relocation, the suitable replacement is found based on proximity and degree of neighboring nodes. If more than one neighboring nodes are found at same distance from failed node and have same degrees, then node ID is used for final arbitration. DARA is applicable for finding single node failures. Authors also propose two variants of DARA namely DARA-1C and DARA-2C to address 1 and 2-connectivity requirements respectively [24]. The main idea is to find least number of sensor nodes that need to relocate to reestablish k-level of connectivity. In DARA-1C, the relocation of neighboring node is recursively applied in order to handle the connectivity breakage due to the movement of one of their neighbors. DARA-1C is further extended to restore 2-connectivity. Similarly, in DARA-2C, first nodes that are affected and have lost their 2-connectivity property are identified. Some of these nodes are then relocated in order to restore 2-connectivity. Both versions of DARA pursue node relocation and fundamentally differ in the scope of the failure analysis and the recovery. In some cases, DARA enhances the path length and does not focus on coverage issues.

Abbasi et al. [25], considered the problem of DARA, and proposed Least-Disruptive topology Repair (LeDiR) approach. The scheme moves a complete block of nodes towards failed node rather than single node movement. During movement, the block is stretched so that the block can cover the complete black hole created due to failure. The scheme evaluates the position related information of all nodes in the network through depth first search, which is too costly. Wang et al. [26], also consider another problem of DARA. In [26], the author claims by giving some counter examples that DARA will not work smoothly in all scenarios and presented a solution which works in all possible scenarios with minimum movement overhead as compared to DARA.

Akkaya et al. [11], propose two distributed schemes, namely Partition Detection and Recovery Algorithm (PADRA) and Multiple PADRA (MPADRA). The PADRA handles single cut vertex node failure at a time. The author proposes a cascaded dominating set (CDS) based approach in order to find whether a node is cut vertex or not in advance. For handling failure of all possible cut vertices, the PADRA assigns some failure handlers (FHs) in advance to move to the location of failed cut vertices. As PADRA can only handle single node failure at a time, another scheme MPADRA is proposed for multiple simultaneous failures. It is reported that schemes are able to handle failure efficiently with less movement. But, in both schemes, the impact of failure of nodes on coverage is not analyzed and is also not considered during failure handling. In order to reconstruct network topology in sensor actor network, Ranga et al. [27] propose a distributed prioritized connectivity restoration algorithm (DPCRA) based on a timer. The cut-vertices nodes are found through DFS search as in PADRA and are assigned some priority based FHs. Whenever an FH detects failure of concern cut-vertex node, it waits for its time and initiates the recovery process if it has not already been initiated by any other high priority FH. The main drawback of the scheme is the longer recovery time as compared to other schemes such as DARA, PADRA, etc. Similar to DARA and PADRA, Guizhen et al. [28] propose a cut-vertex failure recovery scheme aiming to minimize the communication overhead. The approach reduces the communication overhead by reducing cut-vertex searching and information maintenance cost. The algorithm believes on local critical nodes rather than finding global cut-vertices and recover their failure locally.

Younis et al. [9], propose a similar approach as DARA and PADRA called Restoring Connectivity through Inward Motion (RIM). Both DARA and PADRA maintain the information of nodes' 2-hop neighbors and confirm every time, whether failed node is cut vertex or not. While RIM maintains the list of only 1-hop neighbors and handle the failures of all nodes whether cut vertex or not. On failure of a node, RIM handles the failure by moving all 1-hop neighbors towards the position of failed node till they do not form a connected network. The movement of all neighboring nodes towards failed node causes lots of message overhead because every moving node broadcasts a message to its neighbor before it moves from its position. Also, in some cases, the total distance and number of nodes moved is comparatively much higher than other schemes like DARA and PADRA. In order to tolerate failures, Distributed Fault-tolerant Clustering and Routing (DFCR) is proposed by Azharuddin et al. [29]. The cluster head (CH) here is selected on the basis of a cost function which includes remaining energy and proximity of nodes. The algorithm also presents a fault tolerant routing algorithm for handling the sudden failure of a CH without any redeployment and re-clustering.

Tamboli et al. [30] propose a novel coverage conscious connectivity restoration (C^3R) algorithm. The scheme considers both connectivity and coverage issues. C^3R alternatively places neighbors of failed node one by one at the position of failed node according to a defined schedule. Every participating neighbor of failed node moves to the location of failed node, serves for its turn and goes back to its original position. But due to alternative schedule the scheme consumes significant energy in movement. The connectivity between disjoint segments of a network is also reconstructed through relay node placement [31, 32].

In Lee et al. [31] propose a Connectivity Restoration with Assured Fault Tolerance (CRAFT) scheme to establish 2-connectivity between partitioned segments of the network by deploying least number of relay nodes (RNs). Initially, the scheme strives to form the largest inner simple cycle or backbone polygon (BP) around the center of damaged area where no segment lies inside. Then RNs are deployed to connect each outer segment to the BP through two non-overlapping paths. Similarly, two RN based approaches are proposed by Senturk et al. [32] in order to restore the connectivity amongst disjoint segments of a network. The first approach is based on magnetic repelling force applied by RNs and other nodes. A group of relay nodes are placed in between disjoint segments which stretch gradually towards disjoint segment because of repelling forces applied by RNs to each other. RNs repel each other until disjoint segments connect. In the game theory based approach, segments are connected based on priority determined by a leader RN by using probability distribution function (pdf). Partitions with higher pdf are connected first as compared to partitions with lower pdf. In CRAFT, static RNs are placed on calculated locations while in case of [32] mobile RNs are used which relocates themselves after their deployment.

Most of approaches in literature only take connectivity issue while dealing with node failures and ignore sensing coverage. WSNs being densely deployed networks have overlapping sensing and communicating regions due to numerous nearby nodes. Hence, there are chances that failed node's sensing coverage is overlapping with some nearby neighboring node's coverage. In such cases, if neighbor node is back-up node, it can be turned-on to cover failed node's sensing region and thereby avoiding the movement of a mobile node from some remote location. Otherwise, the coverage can be recovered by exploiting movement.

Also, most of the available schemes diagnose failures through time bound heartbeat messages. But, the heartbeat messages may be lost due to many reasons like congestion, collision and loss in channel in case of unreliable wireless channel leading to wrong detection of healthy node as failed one. Therefore, a reliable mechanism for failure identification is needed for avoiding the unnecessary movements in case of false detection of failures. Hence, in this paper, we propose an efficient mechanism to recover network from failures by reducing false alarm rate and utilizing the redundant nodes effectively in such a way that the network always provide k-connectivity for $k \geq 1$ and maintained an adequate coverage during the lifetime of the network.

3 System model

3.1 Network model and assumptions

As shown in Fig. 1, present work assumes a densely deployed WSN comprising of static and mobile sensor nodes. The network is divided into zones of similar sizes where the size of every zone is bounded with communication range. Sensor nodes are deployed randomly in Region of Interest (ROI) where some nodes overlap their sensing region. The network has following assumptions.

(1) The WSN is divided into various zones of similar size, where every sensor node in the network is aware of its zone.

(2) Nodes may suffer crash failures i.e. they are not able to communicate.

(3) The network has single fault free sink node which collects data from all zones through any flat or hierarchical data dissemination protocol.

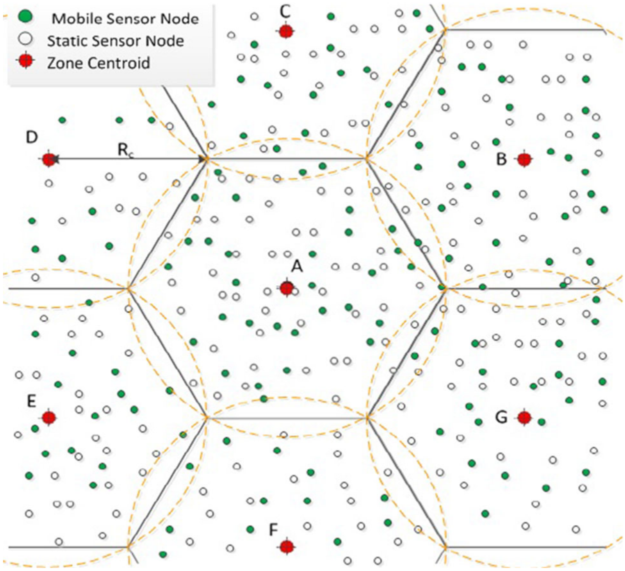

Fig. 1 Illustrations of various virtual zones in a typical sensor network

(4) In order to enhance lifetime, the overlapping sensor nodes follows a turn on/off schedule. The nodes are considered either in ACTIVE (can sense and communicate) or SLEEP (a promiscuous mode where nodes can listen) states.

(5) Initially, sensor nodes in every zone are scheduled in such a way that they provide coverage greater than or equal to an acceptable coverage threshold by turning-on minimum number of sensor nodes.

(6) All sensor nodes are assumed to have identical communication range R_c and sensing range R_s.

(7) Though, some mobile sensor nodes have the ability to move, but they are not considered moving most of the time as in Mobile Ad Hoc Networks (MANETs). Additionally, there is no obstacle on the path of a moving node and the nodes can reach to their exact locations by maintaining a constant speed.

Various notations used in paper and their meanings are given in Table 1 in "Appendix".

3.2 Coverage and connectivity model

To increase coverage using random deployment, node density is kept high by deploying certain redundant sensor nodes. However, increasing nodes density beyond certain threshold does not significantly increase the coverage. This is due to the fact that newly added redundant nodes necessarily do not always occupy the uncovered areas or black holes [33]. Hence, it may happen that certain areas are left uncovered even after deploying nodes with very high density. If network has some mobile nodes, then these nodes can be moved to these uncovered areas to cover the hole. But if there are large numbers of holes created in the network and mobile nodes are relatively few, then many holes may still be left uncovered even after moving these nodes. Also, it is neither desirable nor efficient to initiate mobile node relocation irrespective of the size of the hole because in practical scenarios 100 % coverage is rarely required. Therefore, holes with sizes smaller than a particular size can be left uncovered without affecting network operation and only holes with large sizes should be taken care of. This necessitates only keeping few sensor nodes in ACTIVE mode while rest of the nodes may be kept in SLEEP mode. These redundant nodes kept in SLEEP mode help in failure handling and act as an alternate for providing coverage in case their neighbor nodes fail.

In this section some definitions and lemmas are given for guaranteeing connectivity and adequate coverage in the network. The section also models an acceptable black hole and gives lemmas to ensure the desired coverage and connectivity requirements in the presence of these black holes.

Definition 1 Sampling Points Set SPS = $\{p_i,$ *where* $i = 1,2,3,\ldots,N\}$ is a finite set of points within a given ROI and N is sufficiently large such that $Coverage(SPS) \cong Coverage(ROI)$.

Definition 2 A point p is considered covered if $d(p,s_i) < R_s$, where the $d(p,s_i)$ is Euclidian distance of sensor node s_i from p for $\exists i$, $(1 \le i \le N)$ and R_s is the sensing coverage of node s_i. When there are at least k sensors at a distance less than R_s from p then this is called k-coverage for point p. The p is not covered if it is located exactly on or outside the sensing circle of sensor s_i i.e. distance $d(p,s_j) \ge R_s$ where $i \ne j$ [13].

Definition 3 If two sensor nodes s_i and s_j are at a distance less than or equal to R_c i.e. $d(s_i,s_j) \le R_c$, then they are considered as connected i.e. there exist a communication path between them. If there exist at least k-paths amongst sensor nodes in the network then network is said to be k-connected [34].

Definition 4 The Coverage of ROI is the percentage of covered points in SPS. The coverage of an ROI is calculated with the help of sampling points as:

$$Cov(ROI) = \frac{\left|\bigcup_{i}^{N} p_i\right|}{|SPS|} \times 100$$

where, $p_i \in SPS$ and $d(p_i,s_j) < Rs$, $\exists j$, $(1 \le j \le N)$

If SPS is formed per zone, then coverage of a zone Z is:

$$Cov(Z) = \frac{\left|\bigcup_{i}^{N} p_i\right|}{|SPS|} \times 100 \tag{1}$$

Similarly the coverage of some set of nodes can also be calculated.

Definition 5 Neighbor Set of a node s_i is a set of its neighbor nodes denoted as $NS(s_i) = \{s_j,$ where $d(s_j, s_i) < 2R_s$ and $i \ne j\}$.

Definition 6 Covered Black Hole (CBH): Hole is the area within ROI which either due to node(s) failure or due to absence of sensor node is not falling under sensing region of any active sensor node. If an event starting from within the hole can reach at the boundary of ROI without falling under any sensing region of any sensor node, then the hole is called an uncovered hole otherwise it is called as covered black hole. Figure 2 illustrates both of these holes.

Next, we look into the size of the black hole which is crucial for determining the response on its formation. As given earlier, not every hole created is necessarily to be covered. Holes larger than particular size must be covered and others may be left as such if despite their presence coverage and connectivity is still maintained up to certain threshold. In order to work out the limits/threshold certain lemmas are given as follows.

Lemma 1 *For 2-connectivity between three sensor nodes the maximum area of covered black hole is* $\cong 0.16R_s^2$ *where* $R_s = R_c/2$.

Proof Figure 3 shows three sensors placed at a distance R_c from each other with sensing range $R_s = R_c/2$. By the definition of connectivity [35], two sensor nodes cannot communicate if they are apart at a distance greater than communication range R_c from each other. Hence, black hole shown in Fig. 3 is the maximum allowed black hole while three sensors are maintaining 2-connectivity.

Area of black hole region DEF

$$= area\,of\,\Delta ABC - (3 \times area\,of\,region\,DAF)$$

$$= \frac{\sqrt{3}}{4} \times R_c^2 - \frac{3\pi R_s^2}{6}$$

$$= R_s^2 \times \left(\sqrt{3} - \frac{\pi}{2}\right) \cong 0.16R_s^2$$

Lemma 2 *The worst case coverage of the network is* $\cong$*91 %, where maximum allowed covered black hole is* $0.16R_s^2$ *i.e. if 2-connectivity is maintained.*

Proof The worst case coverage with 2-connectivity is when all sensors are deployed in triangular grid as above and every triangular arrangement of sensors has a coverage hole of $0.16R_s^2$ (Lemma 1). Then, the percentage of covered area in every triangular shape like ABC in Fig. 3 is

$$= ((Covered\,area\,of\,triangle - Uncovered\,area\,of\,triangle) \times 100)/Total\,area\,of\,triangle$$

$$= \frac{\left(\frac{\sqrt{3}}{4} \times (2R_s)^2 - 0.16R_s^2\right) \times 100}{\frac{\sqrt{3}}{4} \times (2R_s)^2}$$

$$= \frac{(\sqrt{3} - 0.16) \times 100}{\sqrt{3}} \cong 91$$

If the complete ROI is divided into n such triangular shapes, then the coverage of complete ROI is the average of the coverage of all shapes, which is also 91 %.

Lemma 3 *If a set of sensor nodes S form a covered black hole, then any two nodes* s_i, $s_j \in S$ *(where* $i \ne j$*) are always connected for* $R_c \ge 2R_s$.

Proof If there are k sensor nodes s_i, $\{i = 0,2\ldots k\text{-}1\}$ which are forming the boundary of a black hole then these nodes must be forming a circular chain of sensing regions overlapping in the following manner:

$$Cov(s_{i\%k}) \cap Cov\left(s_{(i+1)\%k}\right) \neq NULL$$

This means that there should be some overlapping of sensing regions between any two successive sensor nodes $s_{i\ \%k}$, $s_{(i+1)\ \%k}$. Also, distance between these two successive sensor nodes will always be less than or equal to $2R_s$. As we know that communication range is greater than or equal to the double of sensing range ($R_c \geq 2R_s$), hence they will always be connected and hence consequently all nodes around the black hole are connected to each other.

4 Proposed scheme

Entire ROI is divided into zones of similar sizes and are considered to have similar resources. Also, procedures used for fault detection and recovery are similar in every zone. Initially, sensor nodes in each zone elect suitable node as a zone monitor (ZM) which coordinates activities within the zone for handling faults and failures. ZM detects faults through timeouts and to avoid false detection it initiates an agreement among few nodes to find correct agreed status. In case a fault is detected, the resulting hole as per definitions given earlier is checked for its effect on network operation in terms of coverage and connectivity. Accordingly, no action is taken if resulting hole is a covered hole and the size is below a threshold value. Otherwise, a recovery procedure is initiated by ZM either by activating available nearby sensor node or by relocating mobile node from some other part of the network.

4.1 Selection of zone monitor

A distributed approach for selecting ZM with in a zone is proposed which is based on the concept of Bully election algorithm [36]. The algorithm is modified by taking node locations and their energy into consideration for assigning priority. Also, the format and type of communicating messages during election process are modified accordingly to suit to the broadcast over wireless channel.

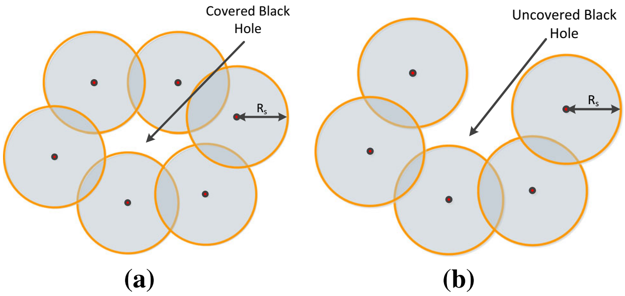

Fig. 2 **a** Covered black hole. **b** Uncovered black hole

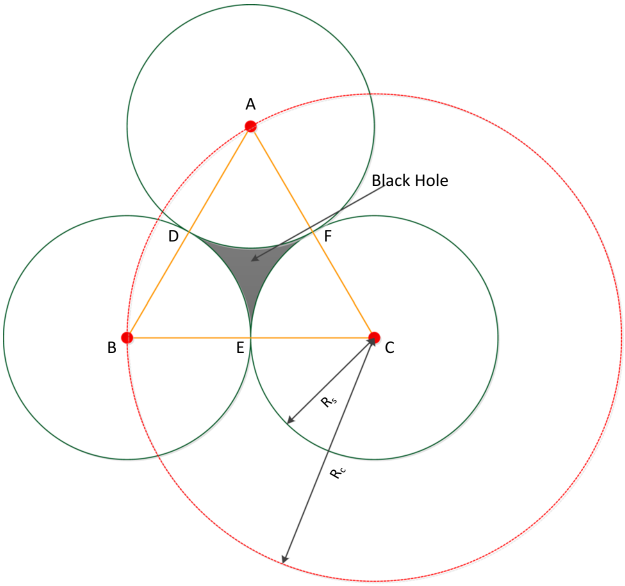

Fig. 3 The maximum allowed black hole in case three nodes have 2-connectivity

4.1.1 Election requirements

Suppose a zone Z has N_{zs} static and N_{zm} mobile sensor nodes where, $N_z = N_{zs} + N_{zm}$ are total number of nodes within the zone. The mobile as well as static node can be elected as a zone monitor. Any sensor node can start the election whenever it detects failure of current ZM. Also, the election can be initiated by the current ZM if it reaches at defined energy threshold. An individual sensor node does not call more than one election at a time, but all N_z individually can call N_z concurrent elections. All nodes are always considered in a promiscuous mode where they can hear the communication. Whenever sensor nodes hear an election message they become active by turning themselves on for the duration of the election. After successful completion of election, if monitor is elected amongst the active nodes, then all other nodes become active or inactive as they were before election. Otherwise the elected node sets itself as active. Each node s_i, $(i = 1, 2, 3,\ldots,N_{zs} + N_{zm})$ has a variable *elected*$_i$, which will contain the identification number (ID) of elected monitor. Whenever a node s_i initiates election, it assigns *elected*$_i$ = "*Undefined*" and becomes a participant. Sensor nodes which are not engaged in any election are called non-participant. Requirements during every run of election algorithm are as follows:

$E1$: ($Safety$) All participant sensor nodes s_i, ($i = 1, 2, 3,\ldots, N_{zs} + N_{zm}$) either have $elected_i =$ "$Undefined$" or $elected_i =$ "ID_{zm}", where ID_{zm} is the unique identification number of non-crashed sensor elected as ZM after the successful completion of election process

$E2$: ($Liveness$) After successful completion of election process, all sensor s_i either set $elected_\mathrm{i} \neq$ "*ID of existing ZM*" or $elected_\mathrm{i} \neq$ "*ID of same alternate node elected with in zone*"

4.1.2 ZM selection criteria

As ZM has extra overhead of coordinating failure identification and recovery activities within the zone, the node elected as a ZM should have comparatively more energy. Also, ZM needs to communicate with other nodes within the zone many times, which creates certain communication overhead for these nodes. The overhead for these nodes depends upon the position of the node elected as ZM. The effect of the position of ZM is shown in Fig. 4. The figure shows that if the size of the zone is bounded by communication range and ZM is located at the centroid A of the zone, then ZM can communicate with all members of zone in one hop i.e. the communication load by ZM on all members is same. But, in case ZM deviates by a distance D from A to B, then some of the members of region 1 are not reachable in one hop from it. For reaching nodes of region 1, some intermediate nodes in region 2 need to be used as relay nodes, which results in extra consumption of their energy causing non-uniform depletion of energy with in a zone. More the area of region 1 more shall be the number of nodes that needs to be reached by ZM using more number of relay nodes from region 2. Hence, to keep the area of region 1 minimum, a node closer to centroid must be selected as ZM. But, node energy being very important factor, ZM selection criteria may also combine available node energy along with above distance from centroid as optimum selection criteria and is here called as monitor priority number (MPN).

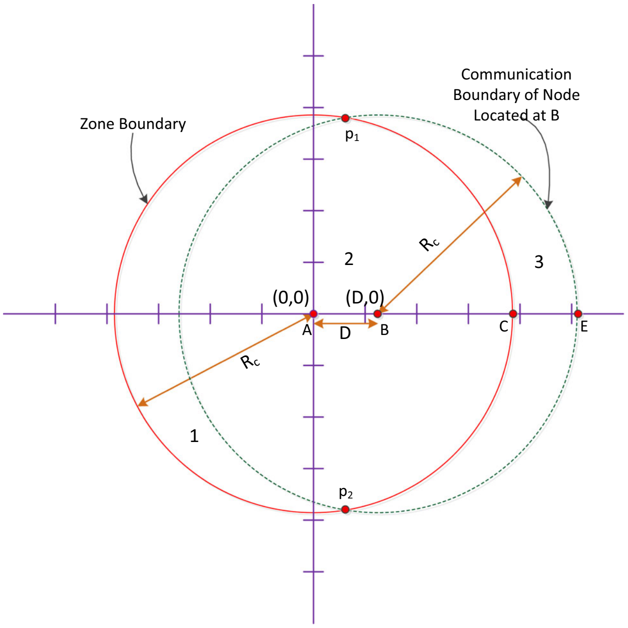

Fig. 4 Impact on connectivity coverage in a zone, if ZM deviates from centroid by a distance D

For election, each node s_i calculates its MPN knowing its remaining energy as well as location with respect to centroid as follows:

$$MPN_i = \frac{1}{(Energy\ spent\ by\ node\ s_i) \times (Number\ of\ nodes\ in\ region\ 1)} \tag{2}$$

where,

$$Energy\ spent\ by\ node\ s_i = Initial\ energy\ of\ node\ (E_0) - Remaining\ energy\ of\ s_i\ (RE_i) \tag{3}$$

Number of nodes in a region where uniform deployment is used can be calculated as:

$$Number\ of\ nodes\ in\ region\ 1 = \frac{N_z}{\pi R_c^2} \times Area\ of\ region\ 1 \tag{4}$$

Although, above number is correct for uniform deployment, but can also be used as an approximation for random deployment. Further, this approximation serves very well the purpose since the number is used for comparison only and not for calculating exact energy needed.

As shown in Fig. 4, the area of region 1 and region 3 is same. Also, the area of region 1 depends upon the distance D from A to B, while the direction of point B will not affect the area of region 1. Therefore, the area of region 3 which is equal to area of region 1 can be calculated as:

$$The\ area\ of\ region\ 3\ (A_c) = 2 \times the\ area\ of\ region\ p_1CE$$

$$A_c = 2 \times \int_0^{\sqrt{R_c^2-(D/2)^2}} \int_{\sqrt{R_c^2-y^2}}^{\sqrt{R_c^2-(y-D)^2}} dx.dy \tag{5}$$

where, coordinates of points p_1, p_2, C, and E are

$$p_1 = \left(\frac{D}{2}, \sqrt{\left(R_c^2 - \frac{D^2}{4}\right)}\right)$$

$$p_2 = \left(\frac{D}{2}, -\sqrt{\left(R_c^2 - \frac{D^2}{4}\right)}\right)$$

$$C = (R_c,\ 0)$$

$$E = (R_c + d,\ 0)$$

From Eq. (5), the area of uncovered region is:

$$A_c = R_c^2\left(\pi - 2\cos^{-1}\left(\frac{D}{2R_c}\right) - \frac{D}{2}\sqrt{\left(4R_c^2 - D^2\right)}\right) \tag{6}$$

Now from Eqs. (3), (4) and (6), the value of MPN_i can be calculated as:

$$MPN_i = \left((E_0 - RE_i) \times \frac{N_z\left[R_c^2\left(\pi - 2\cos^{-1}\left(\frac{D}{2R_c}\right) - \frac{D}{2}\sqrt{\left(4R_c^2 - D^2\right)}\right)\right]}{\pi R_c^2}\right)^{-1} \tag{7}$$

The node with highest MPN amongst non-crashed sensor nodes is always elected as a monitor.

4.1.3 ZM election process

Similar to bully election algorithm, the proposed scheme assumes that message broadcast by a sensor node is delivered to an intended node within a specific time. A maximum turn around time T is the time within which a sensor node can receive the response of the message sent by it. The T can be calculated as:

$$T = 2T_{trans} + T_{proc} \tag{8}$$

where T_{trans} is the maximum estimated time required to propagate a message within a zone and T_{proc} is the maximum estimated processing time required to generate a reply.

The algorithm broadcasts only two types of messages during an election: an election message $MsgE(ID_i{,}MPN_i)$, which is broadcasted when a sensor node s_i initiates an election and a coordinator message $MsgC(ID_j)$, which is broadcasted to all sensor nodes in zone for intimating the ID of elected monitor s_j. In order to balance energy, every elected ZM after its election as monitor defines an energy threshold level. The election process is initiated by the current ZM in case energy of ZM reaches at a defined energy threshold or by other node(s) on detection of failure of current ZM. The complete procedure of election uses two rules: message broadcasting rule and message receiving rule.

According to message broadcasting rule, a node s_i can broadcast an election message *MsgE* in following cases:

(1) If it is a ZM and reaches at its energy threshold.

(2) If it detects the failure of current ZM.

(3) If it already has received a message *MsgE* in current election from some node s_k (where $i \neq k$) which has MPN smaller than itself and has not received any *MsgC* within its waiting time i.e. it is suspecting failure of the node which has earlier started election.

Every node s_i, which wants to broadcast *MsgE*, first calculates its current MPN_i by using Eq. (7), sets variable $elected_i = $ "*Undefined*", and another variable $Wait\_Time_i = \mathrm{T}$. The variable $Wait\_Time_i$ denotes the waiting time for a node s_i after braodcasting an election message. It then broadcast the message $MsgE(ID_i{,}MPN_i)$, in the zone and wait for its waiting time. If node s_i does not receive any message within waiting time then, it assumes that it has highest MPN and broadcasts a message $MsgC(ID_i)$ in order to convey its ID as zone monitor.

Otherwise, on receipt of any message, every node s_i follows the message receiving rule. According to message receiving rules, if a node s_k receivs any message from any other node s_i where $i \neq k$, then it does the following:

(1) If the received message is $MsgC(ID_i)$ then node s_k simply sets its variable $elected_k = $ 'ID_i' and reset variable $Wait\_Time_i = -1$ for indicating that now node s_k is not a participant in any election.

(2) If the received message is $MsgE(ID_i, MPN_i)$, then node s_k compares MPN_i with MPN_k and if the MPN_k is smaller than MPN_i then it waits for a waiting time to either receive ID of elected ZM ($MsgC(ID_i)$) or to initiate new election after the expiration of waiting time.

(3) If on receipt of message $MsgE(ID_i, MPN_i)$, s_k finds that MPN_k is greater than MPN_i, then it immediately broadcasts $MsgE(ID_k{,}MPN_k)$ by applying message broadcasting rule if already has not broadcasted on reciept of any previous message within last waiting time. In case, MPN_k and MPN_i are equal, then tie is resolved by comparing the unique ID of nodes s_k and s_i and node with lower ID number is preferred to break the tie.

The detailed procedure of electing ZM is given in flow chart as Fig. 5. In the figure condition $(ID_i, MPN_i) < (ID_j, MPN_j)$ denotes that first MPN_i is compared with MPN_j and if value of both are found equal then ID_i and ID_j are compared for final arbitration.

4.1.4 Safety and liveness analysis

To achieve safety, more than one sensor node should not be elected as ZM after any successful run of election procedure. In the proposed election algorithm, a sensor node with highest MPN does not receive any election message *MsgE* containing higher MPN than the MPN of itself within its waiting time. Hence, the node with highest MPN is the only node which broadcasts *MsgC* in order to intimate its ID as elected ZM. However, if in rare cases MPNs of two or more nodes are same, tie is resolved by comparing their unique ID so that only one of them can declare itself as ZM and the other quits.

Liveness condition is also satisfied by the proposed scheme. ZM initiates election by broadcasting message *MsgE* within the zone. In case, ZM does not receive *MsgC* with in a bounded time, that means it has highest MPN value and declares itself as monitor. It broadcasts *MsgC* with its own ID so that rest of the nodes of the zone can set their $elected_i$ to it. Otherwise, nodes having higher MPN values further start election by broadcasting *MsgE* with their own MPN values and wait for *MsgC*. This way election continues and terminates when a node with highest MPN is reached and all nodes of the zone set value of $elected_i$ to the same node ID. Hence, the condition of liveness that either all nodes within a zone set $elected_i$ value to the ID of existing ZM or to the same alternate node is met.

4.1.5 Message overhead analysis

Theorem 1 *The worst case message complexity of ZM election algorithm is $O(N_z^2)$.*

Proof The worst case is when ZM reaches at defined energy threshold and all other nodes have MPN greater than MPN of current ZM. In this case, first ZM broadcasts a message *MsgE* in order to initiate election process. If all nodes are not in the reach of current ZM due to communication range limit, the message needs to broadcast $O(N_z)$ times to reach to all nodes within zone. In response to this message, all other nodes broadcast a message *MsgE* because their MPNs are smaller than MPN of ZM. The number of these broadcasted messages is proportional to the number of nodes in a zone (N_z). Again, these messages are relayed by intermediate nodes having smaller MPN than the MPN in receiving message and hence, in worst case the message complexity for broadcasting *MsgE* message is $O(N_z^2)$. Finally, the node with highest MPN will broadcast a message *MsgC* after time *T* for confirmation of its election as ZM. Hence, the total number of messages communicated in this case is $\mathrm{O}(N_z) + \mathrm{O}(N_z^2) + \mathrm{O}(N_z)$, which is $O(N_z^2)$.

Theorem 2 *The best case message complexity of ZM election algorithm is $O(N_z)$*

Proof The best case is when the current ZM reaches at its energy threshold and still has the greatest MPN amongst all sensor nodes in its zone. In that case, ZM first broadcasts the message *MsgE* which require $\mathrm{O(N_z)}$ broadcasts to reach to every node when all nodes are not in communication range of current zone. After a time interval *T*, ZM will broadcast another message *MsgC* to let other nodes know about it with complexity $\mathrm{O(N_z)}$. Hence, the messages complexity is $\mathrm{O(N_z)}$ to complete the election in best case.

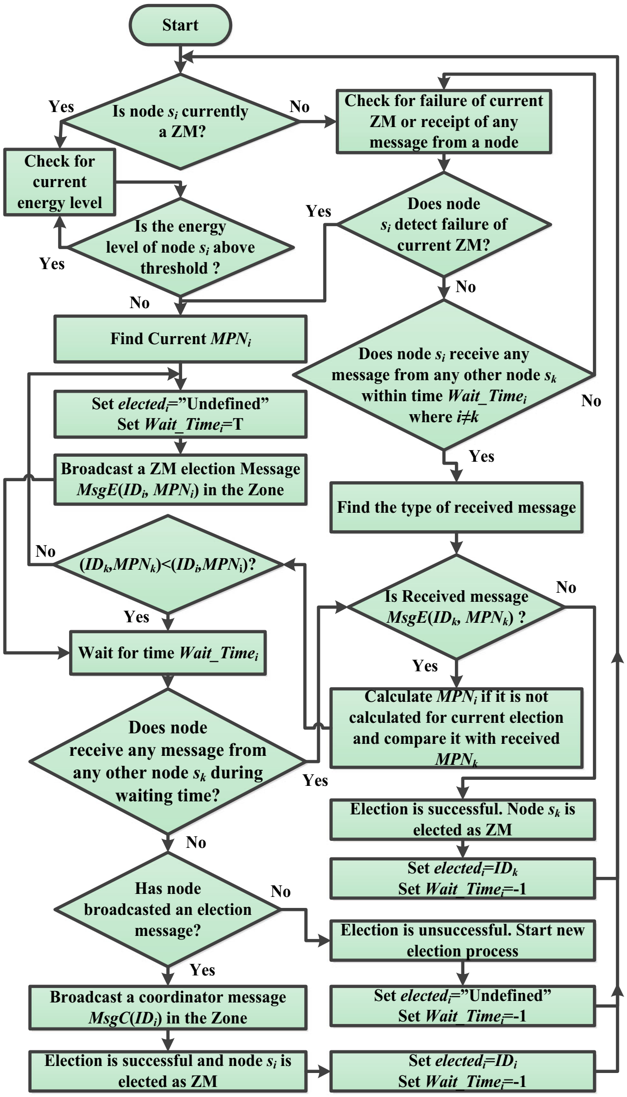
Start
Is node s_i currently a ZM?
Yes
No
Check for failure of current ZM or receipt of any message from a node
Check for current energy level
Is the energy level of node s_i above threshold ?
Yes
No
Does node s_i detect failure of current ZM?
Yes
No
Find Current MPN_i
Does node s_i receive any message from any other node s_k within time $Wait\_Time_i$ where $i \neq k$
No
Yes
Set $elected_i$=”Undefined”
Set $Wait\_Time_i$=T
Broadcast a ZM election Message $MsgE(ID_i, MPN_i)$ in the Zone
Find the type of received message
No
$(ID_k,MPN_k)<(ID_i,MPN_\mathrm{i})$?
Yes
Is Received message $MsgE(ID_k, MPN_k)$?
No
Yes
Wait for time $Wait\_Time_i$
Calculate MPN_i if it is not calculated for current election and compare it with received MPN_k
Does node receive any message from any other node s_k during waiting time?
Yes
Election is successful. Node s_k is elected as ZM
No
Set $elected_i$=ID_k
Set $Wait\_Time_i$=-1
Has node broadcasted an election message?
No
Election is unsuccessful. Start new election process
Yes
Broadcast a coordinator message $MsgC(ID_i)$ in the Zone
Set $elected_i$=”Undefined”
Set $Wait\_Time_i$=-1
Election is successful and node s_i is elected as ZM
Set $elected_i$=ID_i
Set $Wait\_Time_i$=-1

Fig. 5 Flow chart for executing election process by a sensor node s_i

4.2 Failure identification

All active nodes periodically broadcast heartbeat messages in order to inform their aliveness. But, these messages may be lost in wireless channel due to congestion, hindering, etc. Therefore, it is not necessary that ZM and other intended nodes always hear these heartbeat messages successfully. Accordingly, if a heartbeat message is not received from a node s_i within a given time bound, then marking s_i as failed may be a false alarm. Therefore, in order to reduce such false alarms, an agreement based approach is proposed and is as follows.

According to this scheme, the ZM and nodes within particular radius around a node s_i called as agreement region keep listening to the heartbeat messages broadcasted by s_i. Generally, the size of this region is considered smaller than R_c. Figure 6 illustrates the scenario, where a node shown with cross has an agreement region of radius D_{ar}. Whenever heartbeat messages are not received from node s_i, it is marked as suspicious and *ZM* broadcasts an agreement message *MsgA*(ID_i) containing the ID of suspicious node within the zone. This is done to confirm the failure of the suspected node s_i by performing consensus with rest of the nodes of the agreement region. In response to agreement message from ZM, each node s_k of agreement zone responds with message *MsgFR*(ID_k, *Status*, ID_i) where ID_k is the ID of the replier, *Status* is status of node s_i and ID_i is the ID of s_i. Then, on the basis of majority, ZM decides whether the node s_i has failed or not. Above procedure is summarized as follows.

(1) If *ZM* does not receive heartbeat message from a node $s_i \in Z$ ($s_i \neq ZM$) within a defined time bound then ZM broadcasts a message *MsgA*(ID_i) within the zone.

(2) Each node s_k (where $s_k \in Z$, $d_{ik} \leq D_{ar}$) reply with a message *MsgFR*(ID_k, *Status*, ID_i), where d_{ik} is the Euclidian distance between nodes s_k and node $s_{i.}$.The message *MsgFR* contains the identification of sender as ID_k, the status of node s_i as *Status*, the ID of s_i as ID_i. The *Status* is either "*SUSPECIOUS*" or "*OK*".

(3) ZM waits for a bounded time *T* in order to receive replies from each s_k. From total status messages received, it counts the number of messages with the status of s_i as *"SUSPECIOUS"*.

(4) If number of messages with status of s_i as *"SUSPECIOUS"* is greater than half of the total messages received, then node s_i is considered as failed otherwise not.

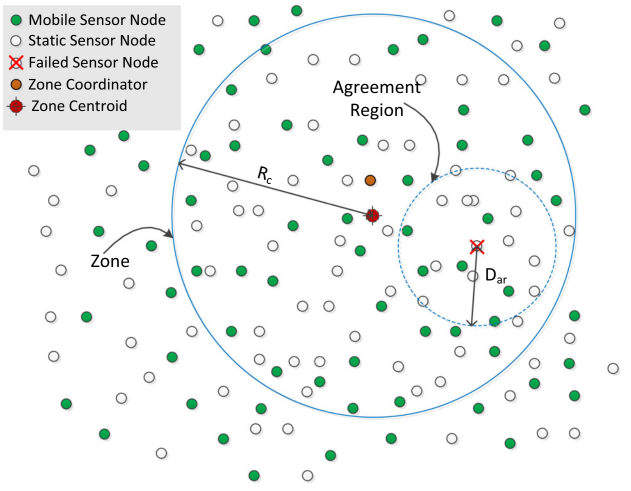

Fig. 6 Illustrations of failure diagnosis within a zone

Theorem 3 *The number of messages required to validate the status of a suspicious node is [*$(\pi R_c D_{ar})^2/N_z + 1$*].*

Proof As shown in Fig. 6, the number of messages broadcasted for confirming the failure status for a suspicious node depends on the size of agreement region. If radius of an agreement region is D_{ar}, then the number of nodes approximately in this zone are $(\pi R_c D_{ar})^2/N_z$. The total number of messages is the sum of the number of messages broadcast by the ZM and number of messages broadcast by the nodes within consensus zone, which is $[(\pi R_c D_{ar})^2/N_z + 1]$.

4.3 Failure recovery

Whenever a node say s_f fails within the zone, then ZM checks the resulting hole for its effect on network operation in terms of coverage and connectivity. Due to this failure following three cases may arise:

Case 1 Holes created by failure of node(s) $s_f \in Z$ do not degrade coverage of zone *Cov*(*Z*) below a threshold coverage C_{th} i.e. *Cov*(*Z*)-*Cov*(s_f) $\geq C_{th}$

Case 2 Holes created by failure of node(s) $s_f \in Z$ degrade coverage of zone *Cov*(*Z*) below a threshold coverage i.e. $Cov(Z) - Cov(s_f) < C_{th}$. But, there exist some node(s) in *SLEEP* mode which can be turned on to *ACTIVE* mode to cover the hole so that resultant coverage of zone is recovered back to the threshold coverage

Case 3 When $Cov(Z) - Cov(s_f) < C_{th}$ and no nodes in SLEEP mode around the hole exist to recover the coverage to the threshold value i.e. $Cov(Z) \geq C_{th}$

In Case 1, since resultant coverage is above the desired threshold coverage, hence ZM does not apply any recovery mechanism. While for Case2, ZM finds backup node (BN) if any available in the neighborhood i.e. *NS*(s_f). Next subsection gives details of designating a node as backup node. If there exists any back up node then ZM simply instructs this node to turn itself on to ACTIVE mode. It may happen that by turning this node on, its coverage overlaps with some existing node's coverage to the extent that this existing node becomes redundant. Hence, that node might be turned off and added to backup nodes' pool. All such nodes are found and added to the pool.

In Case 3, no BN exists in the neighborhood of s_f which can cover the hole created by its failure. Hence, ZM finds the nearby mobile node say MS_i and moves it near to s_f. If the status of MS_i is ACTIVE, then it its movement may leave some uncovered region or black hole. Then, ZM treats this situation as new failure at the location of MS_i and handles it by applying same procedure for failure handling as above. Otherwise, ZM sets status of MS_i simply as ACTIVE and places it at the location of s_f. The complete process of node failure handling is shown with the help of a flow chart in Fig. 7.

4.3.1 Backup node

A sensor node s_i is a backup node (BN) [12] if there exist a set of $NS(s_i)$ such that coverage provided by all nodes in the set minus the coverage of s_i is still above minimum threshold coverage (C_{th}) i.e. $Cov(NS(s_i)) - Cov(s_i) \geq C_{th}$. As shown in Fig. 8, nodes 1, 2, 3, 4 are sensing neighbors of node 5 and if node 5 gets turned-off even then the whole area is covered by nodes 1, 2, 3, 4. Hence, node 5 may be turned off and treated as BN. ZM finds all such BNs and turn them off in order to use them in case of failures occurring elsewhere in the zone. The Lemma 4 can be used for finding BNs in the zone.

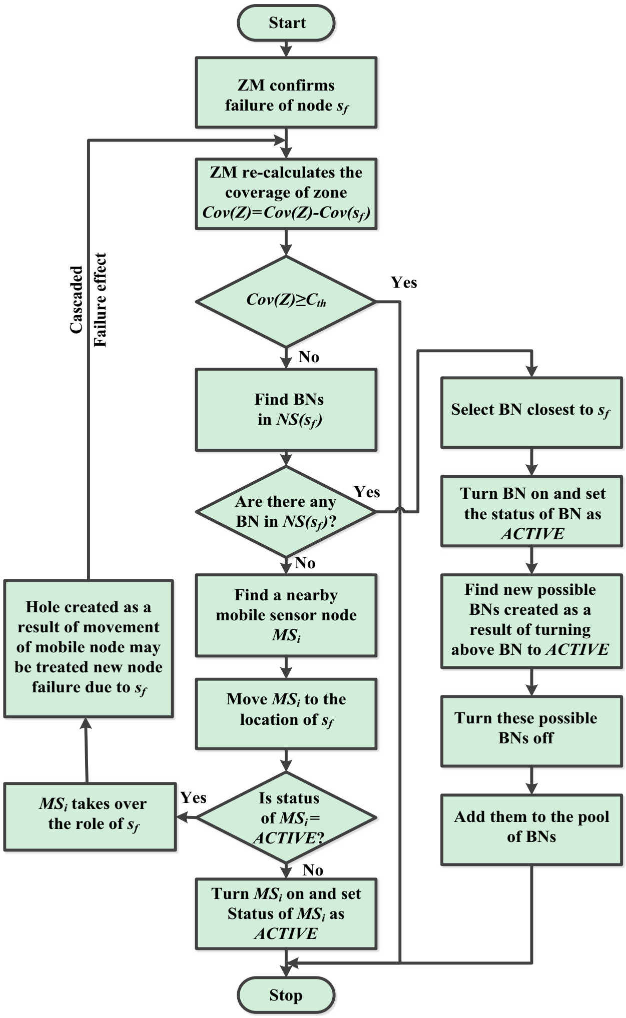
Start
ZM confirms failure of node s_f
ZM re-calculates the coverage of zone $Cov(Z)$=$Cov(Z)$-$Cov(s_f)$
$Cov(Z) \geq C_{th}$
Yes
No
Find BNs in $NS(s_f)$
Are there any BN in $NS(s_f)$?
Yes
No
Find a nearby mobile sensor node MS_i
Move MS_i to the location of s_f
Is status of MS_i= ACTIVE?
Yes
No
Turn MS_i on and set Status of MS_i as ACTIVE
Stop
MS_i takes over the role of s_f
Hole created as a result of movement of mobile node may be treated new node failure due to s_f
Cascaded Failure effect
Select BN closest to s_f
Turn BN on and set the status of BN as ACTIVE
Find new possible BNs created as a result of turning above BN to ACTIVE
Turn these possible BNs off
Add them to the pool of BNs

Fig. 7 Flow chart for failure handling process

Lemma 4 *For a sensor node s_i to be a backup node, $NS(s_i) \geq 3$ i.e. it should at least have three neighbors.*

Proof As given in Definition 7, a node s_i is a backup node if all neighbors $NS(s_i)$ cover its complete coverage area i.e. $Cov(NS(s_i)) - Cov(s_i) = Cov(NS(s_i))$. As shown in Fig. 9, there are two sensor nodes with sensing radius R_s located at O and O' respectively. Then it can be clearly seen that the overlapping area of their sensing coverage decreases as the distance OO' increases and vice versa. The overlapping area is maximum for distance $OO' = 0$ and minimum which is zero for distance $OO' \geq 2R_s$. Similarly, when $0 < OO' \leq 2R_s$, sensing ranges of these two nodes intersect at points A and B and angle θ increases on decrease in distance OO' and decreases on increase in distance OO'. The range of θ for $0 < OO' \leq 2R_s$ can be calculated as:

$$\theta = 2 \times \sin^{-1}\left(\frac{AB}{2R_s}\right)$$

where, $AB = 2 \times \sqrt{\left(R_s^2 - \frac{(OO')^2}{4}\right)}$From above equation, it is clear that for all possible values of AB (for $0 < \mathrm{OO'} \leq 2\mathrm{R_s}$), $\theta < 180°$ i.e. if a rare case where one sensor is lying on top of other sensor is not taken, then in no way a single sensor node can provide hundred percentage coverage for the other node.

Similarly, if second sensor node tries to cover the region of node in hand, it also can provide coverage limited to $\theta < 180°$ only. Therefore, even when combined together both the sensor nodes will provide coverage in the range $0 \leq \theta < 360°$, which means still some region of the sensing region is left uncovered. Then, it is obvious that third sensor node is required to provide complete coverage for the sensing region of a given node.

5 Performance evaluation

5.1 Simulation environment and parameters

The proposed approach is simulated on a widely used network simulator *ns*-2.35 with *mannasim* framework. Simulator *ns* with *mannasim* is used to create wireless sensor nodes of different types and in present experiments we use mica mote2 sensor nodes. A zone of 100 m of radius with varying number of mobile and static sensor nodes is considered. Sensor nodes are deployed in the zone as per normal distribution in order to properly model the actual sensor deployment from an airplane or helicopter.

The communication (R_c) and sensing (R_s) ranges are 100 and 15 m respectively, while the number of nodes with in a

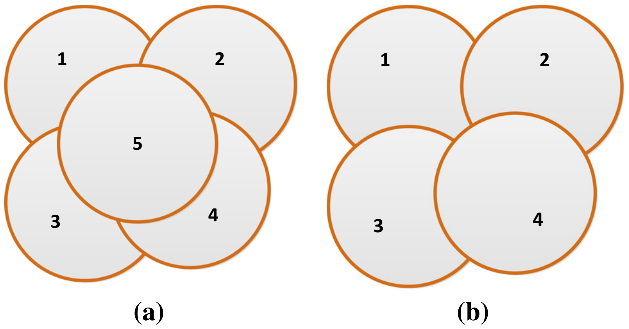

Fig. 8 **a** Illustration of arrangement of five nodes covering certain area. **b** Node *5* is BN because whole area is still covered by nodes *1, 2, 3, 4* after turning-off node *5*

zone are varied from 40 to 180. Performance of proposed scheme is recorded for different values of D_{ar} and the simulation is run for 1500 ms every time. The simulation run for each case is repeated 400 times and average of 400 runs is used to draw results. The proposed scheme is analyzed extensively and results are compared with various failure handling schemes such as PADRA, RIM, and DARA.

5.2 Performance metrics

(1) *False alarm rate (FAR)* It is the percentage of nodes which are identified falsely as failed. The low false alarm rate avoids the unnecessary and undesirable movements of nodes in the network.

(2) *Number of messages exchanged* This is the number of messages exchanged amongst nodes belong to a zone in ZM election, failure detection and recovery processes. This captures communication overhead of the approach.

(3) *Coverage* This is the ratio of total area of the zone covered by sensor nodes in that zone to the total area of the zone. The acceptable coverage in this work is considered approximately 91 % and is called threshold coverage. This metric helps in providing information about coverage degradation in case of failures.

(4) *Total distance moved* It is the total distance moved by sensor nodes to occupy positions of failed nodes. This metric provides zone wise assessment of efficiency of the applied recovery scheme.

(5) *Number of relocated nodes* This is the total number of nodes move from their positions in order to recover failure. This metric also provides zone wise assessment of efficiency of the applied recovery scheme.

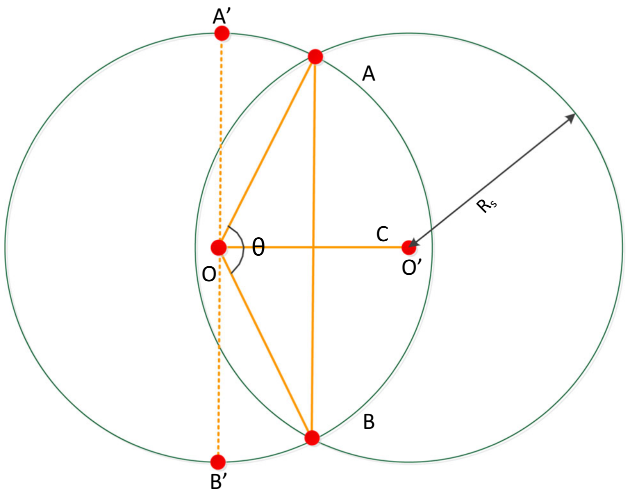

Fig. 9 Illustration coverage overlapping by two sensor nodes

5.2.1 False alarm rate (FAR) analysis

As proposed scheme uses majority based approach for confirming the status of a suspicious node, the false alarm rate (FAR) of the scheme is better than the other schemes. In order to analyze FAR and failure detection accuracy, the scheme is compared with PADRA which uses a heartbeat message based approach in order to detect failed nodes. Failure detection mechanisms used by PADRA, DARA and RIM are fundamentally similar and hence proposed scheme is compared only with PADRA for analyzing FAR and detection accuracy. Results in Fig. 10 reveal that PADRA has FAR approximately 55 % higher than ZBFR. Overall, the FAR increases with increase in node failure rate. In PADRA, if a pre-assigned failure handler node does not receive the heartbeat message from a concerned node within defined interval then it considers that node as failed and executes the recovery mechanism. While in ZBFR, if heartbeat messages are not received from a node then the node initially is considered as suspicious only. Subsequently, ZM initiates a process to collect the status of that suspicious node from its neighboring nodes (nodes of agreement region) and confirms the correct status on majority basis. This reduces the false alarm rate and avoids unnecessary node movements. Figure 10 shows the impact of agreement region on FAR. As the size of agreement region (D_{ar}) increases, the FAR decreases significantly. Results in Fig. 11 shows that fault detection accuracy of both schemes is almost same. This attributes to the fact that in case of a node failure neither a failure handler in PADRA nor neighbors of failed nodes in ZBFR receives the heartbeat messages. Hence, they are able to detect failed nodes most of the time.

5.2.2 *Message overhead analysis*

In proposed approach, messages are mainly exchanged during ZM election and failure detection. In Fig. 12, the massage overhead of proposed failure detection approach is recorded for three different radii of agreement region $D_{ar} = \{R_c/4, R_c/3, R_c/2\}$. Message overhead increases with the increase in number of failures for all different D_{ar}. Also, the overhead is more for larger value of D_{ar}. The increase in overhead with increase in the size of agreement region is due to the fact that as the size of agreement region increases the number of participating nodes increases and the consequently number of messages broadcasted for confirming the status of a suspicious node increases. Figure 13 shows the comparison in number of messages required for electing a zone monitor by proposed ZM election and bully algorithm. In can be seen in figure that the messages overhead of proposed election scheme is less as compare to bully election algorithm.

The total number of messages communicated by proposed scheme is also compared with PADRA, DARA and RIM in Fig. 14. As PADRA designates a failure handler for every node at the time of topology construction or network deployment, it exchanges only few messages during failure handling. In comparison to RIM and DARA, the performance of proposed scheme is comparable to DARA and better than the RIM. In ZBFR, messages are mainly exchanged in ZM election and failure identification. While in case of DARA and RIM all nodes which move in order to recover connectivity, exchange messages with their neighbors before they move from their position. However, the message overhead of ZBFR is more as compared to DARA and PADRA, but the scheme saves much more energy by reducing total number of displaced nodes and distance moved by them to recover failures.

5.2.3 *Coverage analysis*

The change in coverage with increase in number of failures can be seen in Fig. 15. The figure depicts that the proposed scheme is able to maintain the coverage of zone above threshold coverage C_{th} even if 10–15 % of nodes fail. While in case of PADRA, RIM and DARA the coverage decreases sharply for every failure of nodes. This attributes to the usage of few nodes as backup nodes which easily recover the coverage.

5.2.4 *Mobility analysis*

Figure 16, report the distance moved by nodes during failure handling under varying node density. For comparing the proposed approach with similar schemes like DARA, PADRA, and RIM which use mobile nodes, we deploy varying number of mobile nodes within a zone of 100 m of radius. Figure 16 shows that ZBFR performs better than other approaches. With the increment in nodes density the movement is less because the replacement is found easily in the vicinity of failed node. But on the contrary, the performance of RIM degrades sharply with increase in node density. This is due to the fact that in RIM all neighboring nodes of a failed node move and hence higher node density results in higher node movements. However, in ZBFR, DARA and PADRA the total distance moved decreases as the number of nodes increases. The total distance moved in case of ZBFR is nearly 50 and 55 % less than PADRA and DARA respectively.

Results in Fig. 17 show the total number of nodes which are displaced in order to handle failures. RIM displaces more number of nodes as compared to other three schemes and trend is similar as shown in Fig. 16. The number of displaced nodes in case of RIM increases sharply with the

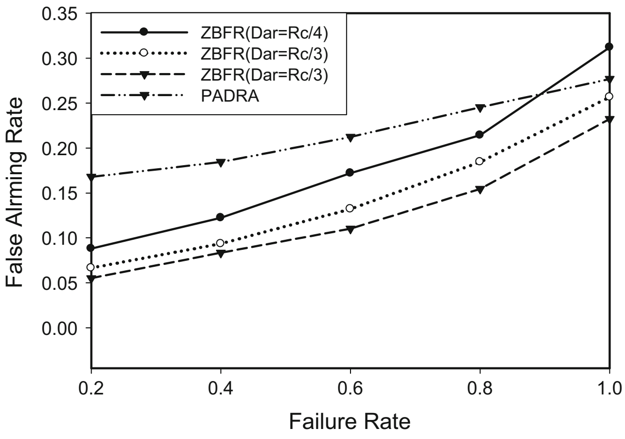

Fig. 10 Illustration of false alarm rate versus failure rate where $N_z = 100$

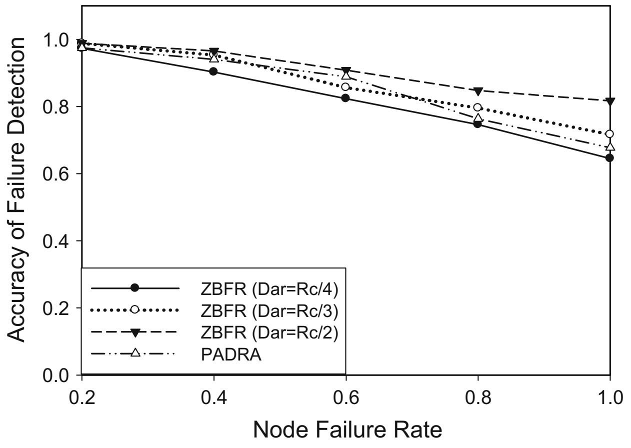

Fig. 11 Illustration of failure detection accuracy versus failure rate where $N_z = 100$

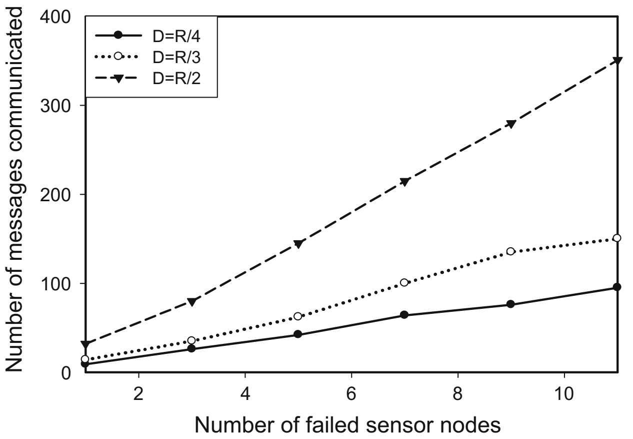

Fig. 12 Illustration of message overhead with the increase of number of failures where $N_z = 100$

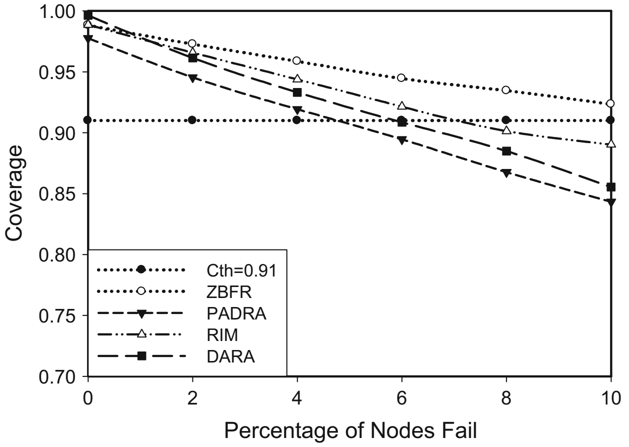

Fig. 15 Illustration of change in coverage with increasing number of failures where $N_z = 100$

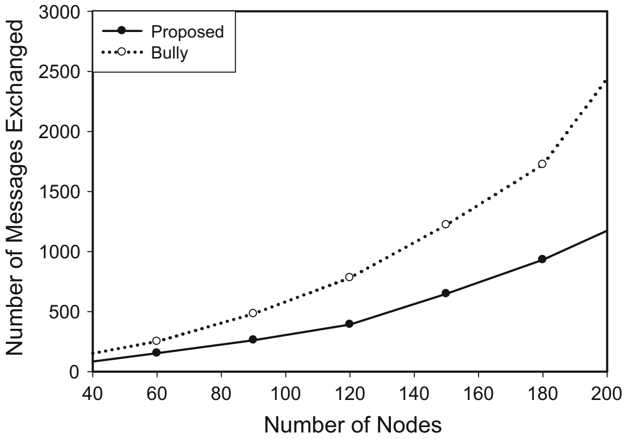

Fig. 13 Number of messages exchanged during ZM election for deferent zone sizes with varying number of nodes

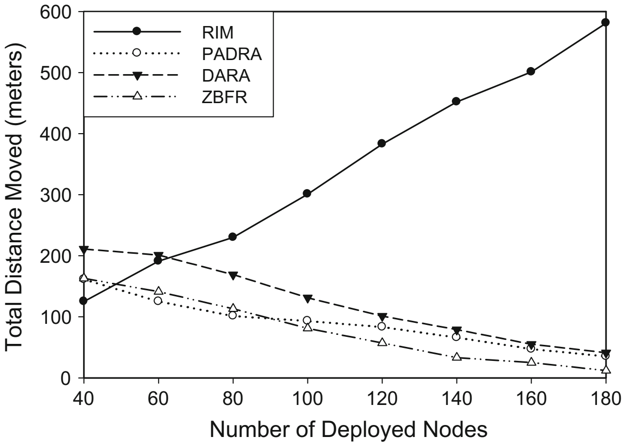

Fig. 16 The comparison of distance moved by nodes during failure recovery in sparse network where the zone size is 100 m

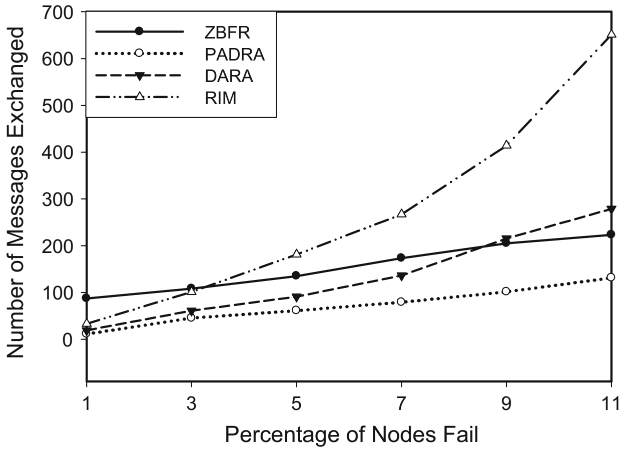

Fig. 14 Comparison of number of messages exchanged where Nz = 100 and Dar = Rc/3

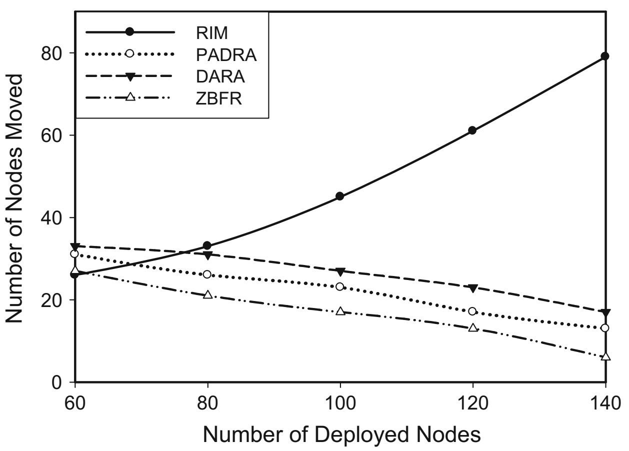

Fig. 17 Number of nodes moved during recovery for different density of nodes where zone size is 100 m

increase in node density. However, amongst ZBFR, PADRA and DARA, the ZBFR seems utilizing mobility more effectively as compared to others and displaces lesser number of nodes.

6 Conclusion

Efficient and timely diagnosis of a failure is very essential in self-configuring and self-healing networks like WSNs. The present work targets at crash fault/failures of nodes and provides a way for handling these failures with reduced false alarm rate. Failures are confirmed by consensus amongst several neighboring nodes of a suspicious node. Proposed scheme thrives to maintain desired sensing coverage in the presence of random faults by utilizing back-up and mobile nodes. Simulation experiments reveal that the proposed failure identification approach is able to diagnose failures by exchanging very less number of massages. Also, the scheme is able to diagnose failures with very low false alarm rate as compared to other schemes. Further, the proposed recovery mechanism is more effective in terms of number of nodes moved while covering the holes. The approach is able to maintain threshold coverage of the network even if 10–15 % of nodes fail.

In future, the scheme can be modified to operate in a fully distributed environment i.e. like consensus amongst nodes needs not to communicate with zone monitor for conclusion. Also, the presence of a centralized zone monitor can be eliminated by locally deciding a failure handler. Consequently, a method to handle nodes' location information in a distributed manner must be worked out.

Appendix

See Table 1.

Table 1 The notations used in the proposed work

Acronym	Explanation
s_i	Sensor node *i*
R_c	Communication range
R_s	Sensing range
$d(s_i,s_j)$	Euclidian distance between any two nodes s_i and s_j
$Cov(X)$	Coverage of *X* where *X* can be a set of sensor nodes or any region
C_{th}	Threshold coverage
$NS(s_i)$	Set of neighboring sensor nodes of s_i
Z	Represents a zone
N_{zs}	Number of static sensor nodes in a zone Z
N_{zm}	Number of mobile sensor nodes in a zone Z
N_z	Total number of nodes in a zone Z
MPN_i	Monitor priority number of a sensor node s_i
E_0	Initial energy of sensor nodes
RE_i	Remaining energy of a sensor node s_i at a particular time
T	A maximum turn around time
T_{trans}	The maximum estimated time required to propagate a message within a zone
T_{proc}	The maximum estimated processing time required to generate a reply
ID_i	Unique identification number of a sensor node s_i
D_{ar}	The radius of agreement zone
MS_i	A mobile sensor node *i*
Z_r	Radius of a zone Z
Status	Current status of a sensor node which can be either *"SUSPECIOUS"* or "*OK*"
$MsgE(ID_i,MPN_i)$	Message broadcasted by s_i to initiate election
$MsgC(ID_i)$	Message broadcasted by $\mathrm{s_i}$ for intimating the IDof elected monitor
$MsgA(ID_i)$	An agreement message broadcasted by zone monitor in order to confirm the status of a suspicious node
$MsgFR(ID_k,Status,\ ID_i)$	Message broadcasted by a sensor nodes s_k in agreement zone of a node s_i in response to message $MsgA(ID_i)$

References

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. *Computer Networks, 38*, 393–422.
2. Chong, C. Y., & Kumar, S. P. (2003). Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE, 91*(8), 1247–1256.
3. Jelicic, V., Razov, T., Oletic, D., Kuri, M., & Bilas, V. (2011). Maslinet: A wireless sensor network based environmental monitoring system. In *Proceedings of 34th international convention on information and communication technology, electronics and microelectronics*, pp. 150–155.
4. Naderan, M., Dehghan, M., & Pedram, H. (2009). Mobile object tracking techniques in wireless sensor networks. In *Proceedings of international conference on ultra modern telecommunications*, pp. 1–8.
5. Hajiyev, C., & Caliskan, F. (2013). *Fault diagnosis and reconfiguration in flight control systems* (Vol. 2). Berlin: Springer Science & Business Media.
6. Elhadef, M., Boukerche, A., & Elkadiki, H. A. (2008). Distributed fault identification protocol for wireless and mobile ad hoc networks. *Journal of Parallel and Distributed Computing, 68*(3), 321–335.
7. Venkataraman, G., Emmanuel, S., & Thambipillai, S. (2008). Energy-efficient cluster-based scheme for failure management in sensor networks. *IET Communications, 2*(4), 528–537.
8. Abbasi, A. A., Akkaya, K., & Younis, M. (2007, October). A distributed connectivity restoration algorithm in wireless sensor and actor networks. In *Local computer networks, 2007. LCN 2007. 32nd IEEE conference on,* pp. 496–503, IEEE.
9. Younis, M., Lee, S., & Abbasi, A. A. (2010). A localized algorithm for restoring internode connectivity in networks of moveable sensors. *Computers, IEEE Transactions on, 59*(12), 1669–1682.
10. Younis, M., Sentruk, I. F., Akkaya, K., Lee, S., & Senel, F. (2014). Topology management techniques for tolerating node failures in WSNs: A survey. *Computer Networks, 58*, 254–283.
11. Akkaya, K., Senel, F., Thimmapuram, A., & Uludag, S. (2010). Distributed recovery from network partitioning in movable sensor/actor networks via controlled mobility. *Computers, IEEE Transactions on, 59*(2), 258–271.
12. Tian, D., & Georganas, N. D. (2002, September). A coverage-preserving node scheduling scheme for large wireless sensor networks. In *Proceedings of the 1st ACM international workshop on wireless sensor networks and applications,* ACM, pp. 32–41.
13. Al-Shalabi, A. A., & Manaf, M. (2012, November). DkCS: An efficient dynamic k-coverage scheduling algorithm for Wireless Sensor Networks. In *Telecommunication technologies (ISTT), 2012 international symposium on,* IEEE, pp. 94–99.
14. Paradis, L., & Han, Q. (2007). A survey of fault management in wireless sensor networks. *Journal of Network and Systems Management, 15*(2), 171–190.
15. Mahapatro, A., & Khilar, P. M. (2013). Fault diagnosis in wireless sensor networks: A survey. *Communications Surveys & Tutorials, IEEE, 15*(4), 2000–2026.
16. Lin, G., & Xue, G. (1999). Steiner tree problem with minimum number of steiner points and bounded edge-length. *Information Processing Letters, 69*, 53–57.
17. Cheng, X., Du, D.-Z., Wang, L., & Xu, B. (2008). Relay sensor placement in wireless sensor networks. *Wireless Networks, 14*, 347–355.
18. Chen, D., Du, D.-Z., Hu, X.-D., Lin, G.-H., Wang, L., & Xue, G. (2001). Approximations for Steiner trees with minimum number of Steiner points. *Theoretical Computer Science, 262*(1), 83–99.
19. Li, J., Shatz, S. M., & Kshemkalyani, A. D. (2011). Mobile sampling of sensor field data using controlled broadcast. *IEEE Transactions on Mobile Computing, 10*(6), 881–896.
20. Vaidya, S., & Younis, M. (2010). Efficient failure recovery in wireless sensor networks through active spare designation. In*: Proceedings of the 1st Int'l workshop on interconnections of wireless sensor networks (IWSN'10),* Santa Barbara, California, pp. 1–6.
21. Imran, M., Younis, M., Said, A. M., & Hasbullah, H. (2010, December). Partitioning detection and connectivity restoration algorithm for wireless sensor and actor networks. In *Embedded and ubiquitous computing (EUC), 2010 IEEE/IFIP 8th international conference on,* IEEE, pp. 200–207.
22. Bagci, H., Korpeoglu, I., & Yazıcı, A. (2015). A distributed fault-tolerant topology control algorithm for heterogeneous wireless sensor networks. *Parallel and Distributed Systems, IEEE Transactions on, 26*(4), 914–923.
23. Huang, Y., Martínez, J. F., Díaz, V. H., & Sendra, J. (2014). A novel topology control approach to maintain the node degree in dynamic wireless sensor networks. *Sensors, 14*(3), 4672–4688.
24. Abbasi, A. A., Younis, M., & Akkaya, K. (2009). Movement-assisted connectivity restoration in wireless sensor and actor networks. *Parallel and Distributed Systems, IEEE Transactions on, 20*(9), 1366–1379.
25. Abbasi, A. A., Younis, M. F., & Baroudi, U. A. (2013). Recovering from a node failure in wireless sensor actor networks with minimal topology changes. *IEEE Transaction on Vehicular Technology, 62*(1), 256–271.
26. Wang, S., Mao, X., Tang, S. J., Li, M., Zhao, J., & Dai, G. (2011). On "movement-assisted connectivity restoration in wireless sensor and actor networks". *Parallel and Distributed Systems, IEEE Transactions on, 22*(4), 687–694.
27. Ranga, V., Dave, M., & Verma, A. K. (2014). A hybrid timer based single node failure recovery approach for WSANs. *Wireless Personal Communications, 77*(3), 2155–2182.
28. Guizhen, M., Yang, Y., Xuesong, Q., Zhipeng, G., He, L., & Xiangyue, X. (2014). Distributed connectivity restoration strategy for movable sensor networks. *Communications, China, 11*(13), 156–163.
29. Azharuddin, M., Kuila, P., & Jana, P. K. (2015). Energy efficient fault tolerant clustering and routing algorithms for wireless sensor networks. *Computers & Electrical Engineering, 41*, 177–190.
30. Tamboli, N., & Younis, M. (2010). Coverage-aware connectivity restoration in mobile sensor networks. *Journal of Network and Computer Applications, 33*(4), 363–374.
31. Lee, S., Younis, M., & Lee, M. (2015). Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance. *Ad Hoc Networks, 24*, 1–19.
32. Senturk, I. F., Akkaya, K., & Yilmaz, S. (2014). Relay placement for restoring connectivity in partitioned wireless sensor networks under limited information. *Ad Hoc Networks, 13*, 487–503.
33. Adlakha, S., & Srivastava, M. (2003, March). Critical density thresholds for coverage in wireless sensor networks. In *Wireless communications and networking, 2003. WCNC 2003. 2003 IEEE*, Vol. 3, IEEE, pp. 1615–1620.
34. Zhou, Z., Das, S., & Gupta, H. (2004, October). Connected k-coverage problem in sensor networks. In *Computer communications and networks, 2004. ICCCN 2004. Proceedings of 13th international conference on,* IEEE, pp. 373–378.
35. Baidya, S. S., & Bhattacharyya, C. K. (2012, December). Coverage and connectivity in wireless sensor networks: Their trade-offs. In *Sensing technology (ICST), 2012 sixth international conference on,* IEEE, pp. 353–358.
36. Davidson, S. B., Garcia-Molina, H., & Skeen, D. (1985). Consistency in partitioned networks. *Computing Surveys, 17*(3), 341–370.

Krishna Pal Sharma received his B.Tech degree in Computer Science and Engineering from Uttar Pradesh Technical University, Lucknow in 2005 and M.Tech in Computer Science and Engineering from Graphic Era University, Dehradun (UK), India, in 2010 respectively. Currently, he is working towards his Ph.D. in wireless sensor ad hoc network from the Department of Computer Science and Engineering, National Institute of Technology Hamirpur (HP), India. His research interest includes computer networks and wireless Sensor ad hoc Networks.

Teek Parval Sharma is an Associate Professor at National Institute of Technology, Hamirpur (India). He has done his Ph.D. from Indian Institute of Technology, Roorkee (Electronics and Computer Engineering Department), India, in the area of Wireless Sensor Networks. He has published numerous high quality research papers in International/National journals and conferences, and has also contributed in various books of standard international publishers. His research interest includes distributed systems, wireless sensor networks, mobile ad hoc networks and wireless networks.