


RESEARCH

Open Access



# Smart node relocation (SNR) and connectivity restoration mechanism for wireless sensor networks

Mahmood ul Hassan<sup>1\*</sup> , Khalid Mahmood<sup>2</sup>, Muhammad Kashif Saeed<sup>3</sup>, Shahzad Ali<sup>4</sup>, Safdar Zaman<sup>5</sup>, Amin Al Awady<sup>1</sup> and Muhammad Saqib<sup>6</sup>

\*Correspondence:

mahmood.mscs@gmail.com

<sup>1</sup> Department of Computer Skills, Deanship of Preparatory Year, Najran University, Najran, Kingdom of Saudi Arabia

Full list of author information is available at the end of the article

## Abstract

Node failures are inevitable in wireless sensor networks (WSNs) because sensor nodes in WSNs are miniature and equipped with small and often irreplaceable batteries. Due to battery drainage, sensor nodes can fail at any instance. Moreover, WSNs operate in hostile environments and environmental factors may also contribute to nodes failure. Failure of nodes leads to disruption of inter-node connectivity and might also lead to network partitioning. Failure to communicate with each other and with the base station can compromise the basic operation of the sensor network. For restoration of connectivity, a robust recovery mechanism is required. The existing connectivity restoration mechanisms suffer from shortcomings because they do not focus on energy-efficient operation and coverage-aware mechanisms while performing connectivity restoration. As a result, most of these mechanisms lead to the excessive mobility of nodes, which itself causes the utilization of excessive battery. In this work, we propose a novel technique called smart node relocation (SNR). SNR is capable of detecting and restoring the connectivity caused by either single or multiple node failures. For achieving energy efficiency, SNR relies on transmitting a lesser number of control packets. For achieving the goal of being coverage-aware, it tries to relocate only essential nodes while trying to restore connectivity. By performing extensive simulations, we prove that SNR outperforms the existing approaches concerning multiple performance metrics including but not limited to the total number of packets transmitted, total distance moved for connectivity restoration, the percentage reduction in field coverage.

**Keywords:** Cut-vertex, Failure recovery, Network connectivity, Node relocation, Wireless sensor network, Node failures

## 1 Introduction

Sensor Networks comprise various sensor nodes scattered in a geographic area for monitoring different phenomena. A wide range of sensors can be attached to each sensor node making them capable of monitoring diverse conditions. One of the inherent characteristics of sensor networks is that sensor nodes are not replaceable and they stay unattended for a long period without any human intervention [1]. For this reason, sensor

networks should have self-healing, self-organization, and fault tolerance capabilities for a successful operation [2].

Other than the environmental challenges such as natural disasters, one of the major research challenges for sensor networks originated from the design and characteristics of the sensor node. Sensor nodes are small and have limited resources in processing power and memory. Sensor nodes use a battery as a central power source [3]. One or more sensors can be attached to a sensor node for measuring different phenomena from the environment [2]. Sensor nodes detect desired phenomena in the environment and can transmit this information to other nodes via radio transceivers.

Sensor nodes are often deployed in harsh environments where human access is restricted; therefore, the replacement of node batteries is not possible. After deploying nodes into a network of sensors, those nodes communicate with each other to establish a network. This network coordinates the sensor nodes and transmits the detected information to the end-user [3]. As a sensor network covers a large area, not all sensor nodes are within the transmission range of the end-user. Nodes must rely on multi-hop communications for sending information toward the end user [4–6]. The reliability of the collected information can be increased by increasing the node density in an area. During the period, battery exhaustion of the sensor nodes occurs because of message transmission and information processing.

When the battery of a node is fully depleted, this sensor node is considered a dead node. As network nodes begin to die during the operation of a sensor network, connectivity holes begin to appear within the network. A connectivity hole is referred to as an area where the nodes are no longer connected. Connectivity holes lead to loss of inter-node connectivity, leading to the inability to send the end-users sensed information. The first step is to detect a failure which is very trivial for connectivity restoration. After identification of a failed node, the next step is to notify the nodes adjacent to the failed node about the failed node so that these adjacent nodes can reposition themselves such that the disconnected nodes become connected again [7].

Multiple failed nodes may result in disruption of network connectivity, and the network may develop multiple isolated segments or partitions. This scenario fails the flow of information between the sensor nodes and the end-user terminal. This scenario may compromise the basic operation of the sensor network, and to prevent this, network connectivity must be restored by self-organization among the nodes in the network. One possible solution for connectivity restoration is to deploy redundant nodes in place of dead nodes. However, this solution is often impracticable due to the absence of human intervention. Ideally, the connectivity restoration process should be performed by the existing alive nodes in the network. The recovery mechanism needs to be robust, and the overhead on the sensor nodes involved in the recovery process should be minimal because of the resource constraints.

Failure of nodes in a sensor network results in different types of connectivity problems. This work categorized the connectivity problems into four different cases: (1) cut-vertex failure, (2) end node failure, (3) two cut-vertex node failures, and (4) multiple end node failures. Details of all these cases are provided in Sect. 3.

Dealing with the above cases is a challenging task. This paper proposes a novel connectivity restoration technique called smart node relocation (SNR) and Connectivity

Restoration Mechanism to handle the issues associated with network connectivity. The major focus of our work is to deal with the connectivity restoration problem by proposing a mechanism capable of restoring connectivity using the existing nodes in the network; therefore, there is no need to re-deploy new nodes for restoring connectivity. Moreover, SNR does not substantially reduce the network coverage due to the movement of nodes as observed by the other connectivity restoration techniques. As energy efficiency is one of the major concerns in sensor networks; therefore, SNR does not rely on the exchange of large amounts of messages for its operation. SNR can also detect all the identified four connectivity problems and effectively find the solution for each case.

The rest of the paper is organized as follows. In Sect. 2, we describe the most relevant related work. Section 3 consists of the research method used for our research work, and we also elaborate on the four different cases related to connectivity restoration. Section 4 presents the simulation results, and subsequently, Sect. 5 concludes this paper.

## 2 Related work

Connectivity restoration in wireless sensor networks is an area that has been thoroughly studied by researchers [8–12]. Some solutions are based on relocation on-demand, while other solutions rely on post-deployment relocation. The applications requiring sensor nodes to be deployed over large geographic areas use the aerial deployment of the nodes. Due to this, node density throughout is not uniform, and some areas may have a higher density of nodes than some other areas. To achieve a uniform distribution of nodes, relocation of sensor nodes is desired so that connectivity can be established between the sensor nodes and the end-user, and the coverage area can be maximized.

The connectivity aspect is thoroughly studied in the literature, and several approaches are presented [8–12]. Some works are more focused on maximizing the coverage of nodes without affecting connectivity. In [13], the authors considered robot networks for the process of connectivity restoration. A 2-connected network concept is introduced, meaning that there should exist a minimum of two pathways among each pair of nodes in the network. This approach achieves 2-degree connectivity. For dealing with a node's failure, the algorithm strives to achieve 2-connectivity by moving a pair of sensor nodes. In this way, connectivity is restored. In [14], the authors proposed a technique called  $C^2AP$ . In this technique, post-deployment of nodes is used for improving coverage and connectivity. A hierarchical architecture is proposed by the authors called COCOLA in [15], where coverage is maximized without forwarding data path to 1-tier node by the incremental relocation of higher-tier nodes. However, both the proposed solutions,  $C^2AP$  and COCOLA, are incapable of dealing with the failed nodes' implications. In [16], a solution based on the node's cascaded movement is introduced for connectivity restoration due to failed nodes. According to this technique, a nearby node replaces a failed node, which is then replaced by another node, and this process continues until finding a redundant node. In [17], the authors proposed a new method called DARA. DARA uses a scheme based on probability for detecting cut vertices and selecting an appropriate neighbor node to the failed node for relocation. The appropriateness of the neighbor is decided based on the number of communication links.

Cascade movement of nodes is observed in most recent connectivity restoration techniques, but it does not care about the sink node and load distribution role. Therefore, a

sink-oriented cascade model is introduced in a Memetic Algorithm for Topology Optimization against cascading failures (MA-TOSCA) [18]. It helps wireless sensor networks to avoid cascading failures using topology optimization. A local search operator is used on this new network paired metric. As a result, it enhances the network robustness and takes less time as compared to other techniques. Furthermore, this algorithm does not consider the impact of SINR (Signal to interference plus noise ratio), which relates to communication links. Therefore, it is an unaware channel algorithm that affects its overall performance.

Energy-Aware Connectivity Restoration Mechanism for Cyber-Physical Systems of Networked Sensors and Robots [19] consists of three algorithms, i.e. CoRFL, CoRFL2, and CoRFLN. It restores connectivity while taking care of network lifetime, energy level, network lifetime, and ecological conditions. Nodes movement is controlled using a distributed algorithm based on the fuzzy logic, residual energy, node rank, and distance. CoRFL handles the issue if the most qualified node is a cut-vertex. In this case, it will not move, and it asks even any far node to move forward. CoRFL2 and CoRFLN are used to solve the movement of cut-vertex nodes. The supervisor gathers the information of partition nodes. The best recovery node is chosen by using the fuzzy logic system from the live nodes. The supervisor administers cascade movement in this way that it is connected to other partitions. A proper mechanism of coordination among the nodes is used. In CoRFLN, the nearest node is considered standby, whereas CoRFL2 attempts to find its substitute by keeping in view the distance and residual energy.

Most current research on cascading failure of WSNs deals with the single sink network, and a few use multi-sink networks. MA-MSP is a memetic algorithm that supports WSNs struggle cascading failures by using multi-sink placement optimization. In this technique, a local search operator is intended based on a new network balancing metric. This proposed cascade model adequately characterizes the process of cascading of multi-sink in the wireless sensor networks [20].

Efficient Solution for Connectivity Restoration (ESCR) [21] is an energy-efficient technique for the connectivity of wireless sensor networks. The technique restores the network with an efficient consumption of residual energy and slightest node movement. Only such nodes can participate in the network restoration process near the faulty node and have more energy. As a result, unnecessary cascade movement of the nodes during the restoration process is stopped. Moreover, it ensures that the node that participates in the network restoration process has a sufficient energy level not to be exhausted during the process. ESCR consists of two algorithms, i.e., Assigning Backup Nodes (ABN) and Connectivity Restoration Process (CPR). ABN algorithm is used to assign the backup nodes for every node according to their residual energy. This process will be done at the start of the network and repeats in any node failures. CPR is the second algorithm whose fundamental task is to restore the network. In case of any nodes failure, its backup node is moved forward to participate in this process. ESCR compared with other well-known techniques and was found better than others. It is evaluated in an environment where sensor nodes are stationary and only actor nodes can move. Its result can be varied if we consider the mobility of sensor nodes as well.

Geometric Skeleton-based Reconnection (GSR) is proposed in [22]. GSR employs a geometrical skeleton-based approach to logically partition networks into different

segments. A group of nodes having maximum connectivity becomes the geometrical skeleton backbone. Each segment keeps track of all skeletal backbone nodes because it plays an important part in network partitioning. In the case of network partitioning, each segment tries to join the geometrical skeleton backbone. This process leads to the restoration of connectivity. However, GSR assumes that each node knows the locations of all other nodes in the network. Second, it is also a prerequisite that all nodes in the network must be aware of all nodes present in the geometrical skeleton backbone. These assumptions are impractical, particularly for large networks, because keeping all this information in a network with mobile nodes can cause massive overhead. Another problem that may arise during the network's operation is that the skeleton backbone may exhaust its energy soon, causing a decrease in such nodes. After a while, the lack of presence of such nodes may result in compromising the recovery mechanism.

An energy-efficient technique, Intelligent On-Demand Connectivity Restoration for wireless sensor networks (IDCRWSN) [23], has been presented that efficiently uses the sensor nodes' residual energy. IDCRWSN restores the connectivity through redundant nodes, which are managed by Slave Keeper nodes. The Slave Keeper nodes are managed and controlled by the Master Keeper nodes. The Permanent Relocation Algorithm for Centralized Actor Recovery (PRACAR) and Self-Route Recovery Algorithm (SRRA) [24] addresses the connectivity restoration of failed actor nodes. The PRACAR restores failed actor nodes' connectivity, and SRRA provides an optimal path to the relocated sensor nodes.

All the above-mentioned works do not consider connectivity, coverage, and energy efficiency collectively. Our proposed work can be distinguished from the abovementioned works because it addresses connectivity restoration, better coverage, and efficient utilization of energy in an integrated manner.

### 3 Research method

We present in the proposed work the different cases that may come up if single or multiple nodes in a network die. Our work aims to propose a solution capable of recognizing all of the identified cases and then taking necessary actions to restore connectivity. For our proposed algorithms, we assume that all sensor nodes are randomly deployed in the deployment area. After the deployment, all nodes discover their neighbors by exchanging HELLO beacon messages. For the initial relocation of nodes, the mechanism used in [17] is used. Algorithm 1 presents the steps that are used for initial node relocation. Figure 1 shows the initial relocation scenario. It is assumed that all network nodes are homogenous and have the same processing and communications capabilities. For each node, it is assumed that the sensing range and communication range are the same. In this paper, we use  $R_c$  to represent the sensing and communication range.

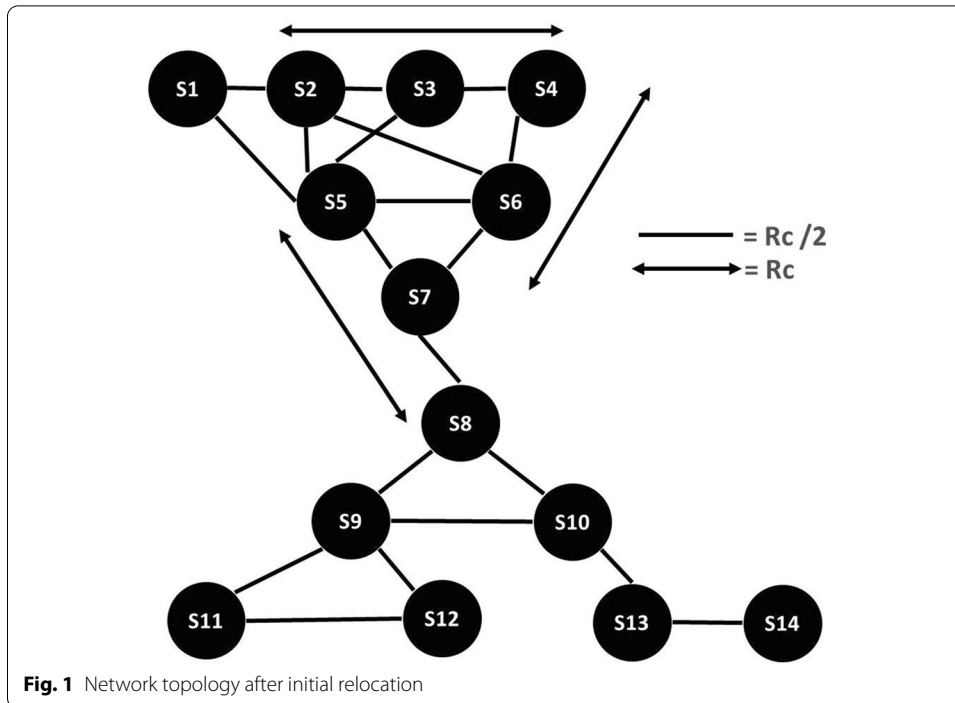


Fig. 1 Network topology after initial relocation

**Algorithm 1:** Relocation( $S, N, \gamma, \emptyset$ )

```

Input:  $S$  ... Set of Sensor Nodes:  $\{s_i\}$ 
Input:  $N$  ... Set of Neighboring Nodes:  $\{n_i\}$ 
Input:  $\gamma$  ... Scalar value as communication range
Input:  $\emptyset$  ... Sink node:  $\emptyset \in S$ 
Output: Initial node relocation successfully done
1. foreach  $s_i \in S$  do
2.    $\Delta s \leftarrow |\emptyset - s_i|$  //node's distance to Sink
3.   if  $\Delta s \geq \gamma/2$  &&  $\Delta s < \gamma$  then
4.     while  $\Delta s > \gamma/2$  do
5.        $s_i \leftarrow s_i + \Delta s$  //keep moving towards Sink
6.        $\Delta s \leftarrow |\emptyset - s_i|$  //update distance to Sink
7.       //notify position to neighbors via HEARTBEAT message
8.     foreach  $n_i \in N$  do
9.        $\Delta n \leftarrow |n_i - s_i|$  //node's distance to Neighbour
10.      if  $\Delta n > \gamma/2$  then
11.        while  $\Delta n > \gamma/2$  do
12.           $s_i \leftarrow s_i + \Delta n$  //keep moving towards Neighbour
13.           $\Delta n \leftarrow |n_i - s_i|$  //update distance to Neighbour
14.          // notify position to neighbors via HEARTBEAT message
    
```

Once the nodes are deployed in an area, each node will broadcast HELLO beacon messages with a transmission range of  $Rc/2$  for providing its location information to other nodes in the network [17]. Each node share information, including its ID and current position in acknowledgment (ACK), to all the neighboring nodes. The transmission range of  $Rc/2$  is used for the transmission of ACK. Also, each node periodically sends a broadcast message for synchronization called SYN message. The transmission range used for transmitting SYN messages is also  $Rc/2$ . SYN messages are used for the

identification of failed nodes. For example, let us consider the scenario presented in Fig. 1. If node S7 has failed, then nodes S5, S6, and S8 will not receive SYN messages from S7. The absence of a SYN message means the failure of a node. Upon detecting a failed node, nodes S5, S6, and S8 will send a HELLO message with a transmission range  $R_c$  toward the failed node direction. Upon receiving the HELLO message, in reply, each node transmits an ACK message. In this way, each node updates the list of neighbors, and they initiate mobility to restore the network, as depicted in Fig. 2.

Upon detecting the failure, it is important to understand the impact of failure on the network topology. In this work, we have categorized four scenarios that can occur due to single or multiple node failures.

### 3.1 Scenario 1: single cut-vertex failure

The cut-vertex scenario is illustrated in Fig. 2. This happens when a node's failure divides the connected network into multiple disjoint partitions [21]. In Fig. 2, it can be observed that the failure of node F divides the network into two partitions resulting in cut-vertex failure. Failure of node F is detected by nodes S5, S6, and S8 due to the absence of SYN messages from node F. Algorithm 2 illustrates the single cut-vertex failure detection and restoration process. For connectivity restoration by our proposed algorithm, nodes S5, S6, and S8 send a broadcast HELLO (also known as Heartbeat) message with a communication range of  $R_c$ . If each node receives an ACK from another neighboring node, the node's mobility needs to restore network connectivity. Upon receiving the ACK from neighboring nodes, and all nodes update the routing list. The solutions presented in [17, 18, 22] rely on the cascaded relocation of neighboring nodes in the given scenario. It is proven that cascaded relocation of nodes leads to more energy consumption leading to quick drainage of sensor

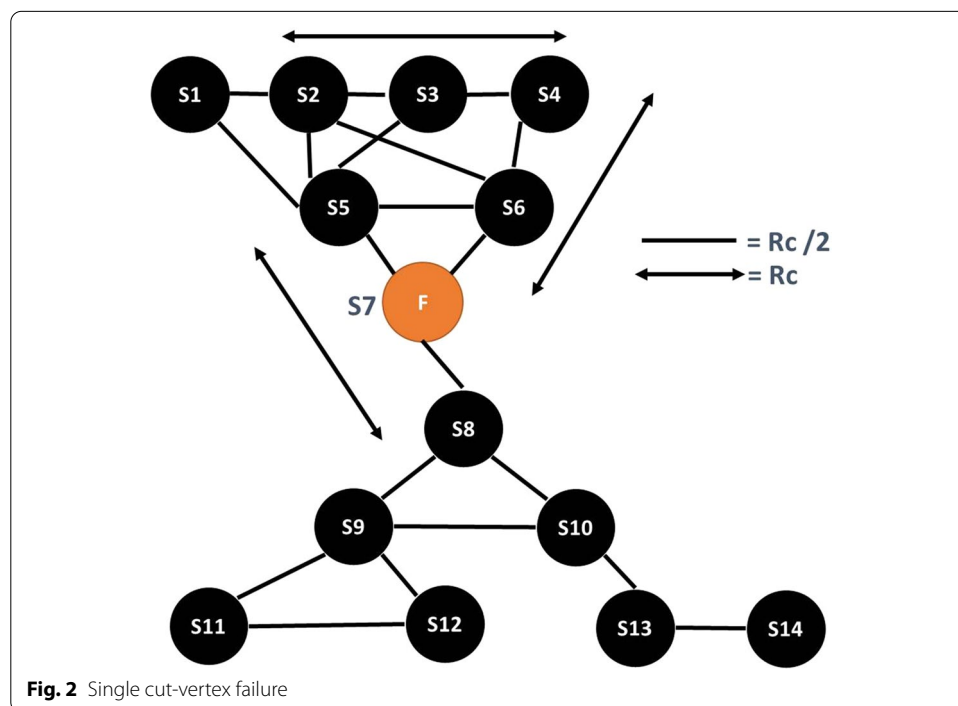


Fig. 2 Single cut-vertex failure

node's batteries [4]. Cascaded relocation also leads to the shrinking of network coverage. Our proposed algorithm improves energy efficiency by avoiding the unnecessary mobility of neighboring nodes and improving network coverage. Essentially, this algorithm also prefers coverage; therefore, these coverage holes will be fulfilled by the neighbors by measuring overlapped distance according to [19].

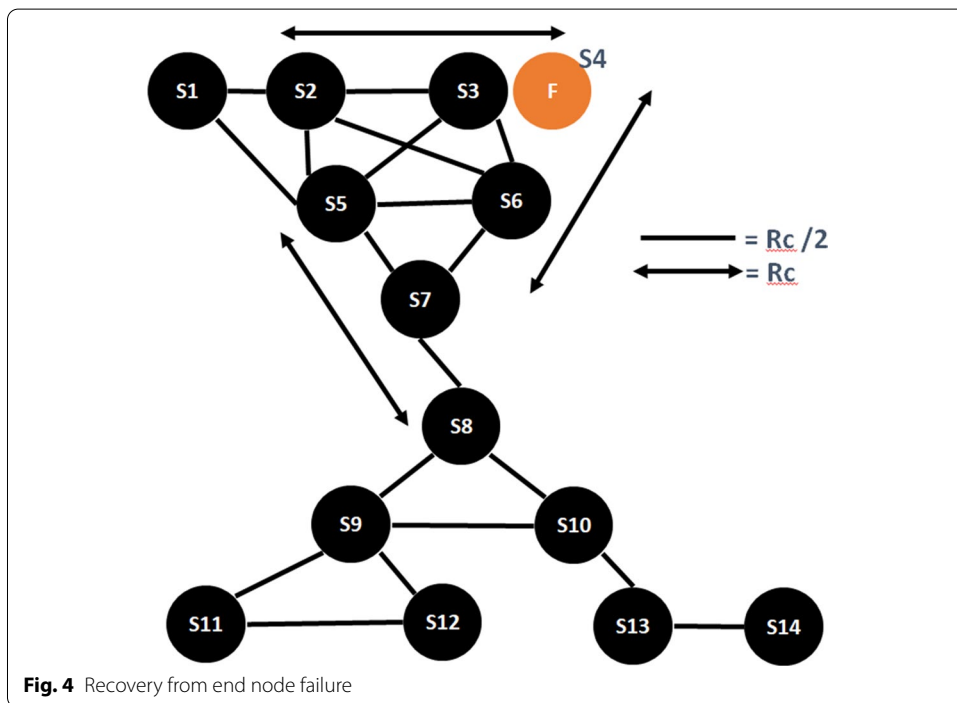
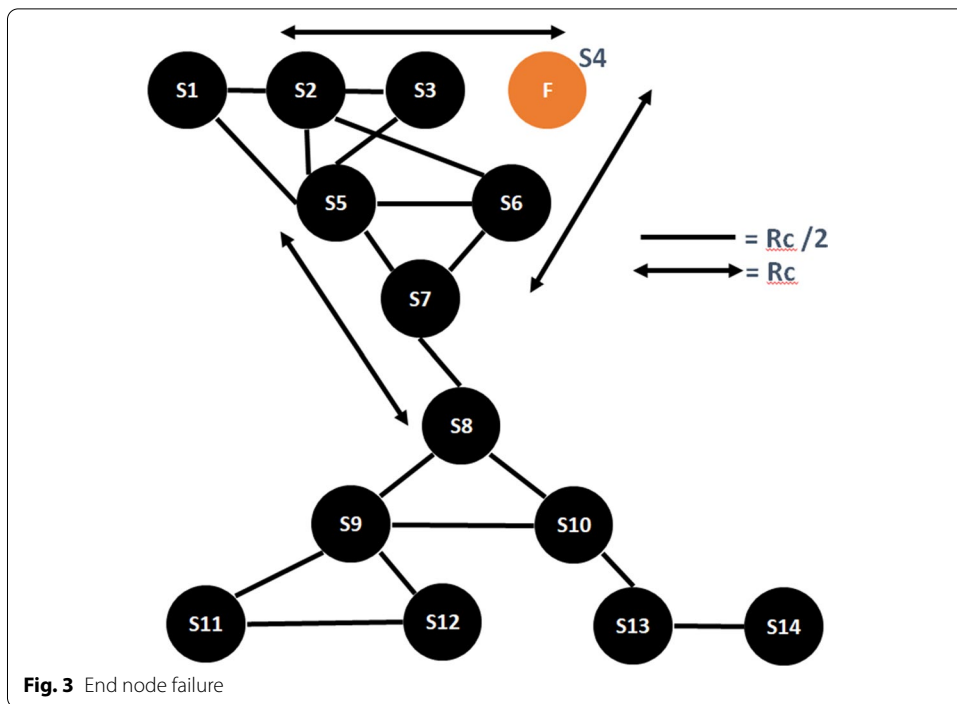
**Algorithm 2:** SingleCutVFDR( $S, C, N, \gamma, \alpha$ )

<p><b>Input:</b> <math>S</math> ... Set of Sensor Nodes: <math>\{s_i\}</math>  <b>Input:</b> <math>C</math> ... Set of Cut Vertex Nodes: <math>\{c_i\}</math>  <b>Input:</b> <math>N</math> ... Set of Neighboring Nodes: <math>\{n_i\}</math>  <b>Input:</b> <math>\gamma</math> ... Scalar value as communication range  <b>Input:</b> <math>\alpha</math> ... Acknowledgement  <b>Output:</b> Single cut-vertex failure successfully detected and restored</p> <ol style="list-style-type: none"> <li>1. <i>foreach</i> <math>c_i \in C</math> <i>do</i></li> <li>2.     <math>n_i \leftarrow \text{Neighbour}(N, c_i)</math>     //ith cut-vertex node's neighbour</li> <li>3.     <i>if</i> <math>n_i</math> detects that <math>c_i</math> has failed <i>then</i></li> <li>4.         <math>n_i</math> broadcast HEARTBEAT with a range equals to <math>\gamma</math></li> <li>5.         <i>if</i> <math>\alpha \leftarrow \text{GetAcknowledgement}(c_i)</math> <i>then</i></li> <li>6.             Update_Routing_Table (<math>n_i</math>);</li> <li>7.         <i>else</i>     //No acknowledgment received</li> <li>8.             Cascaded_Movement();</li> <li>9.             Update_Routing_Table (<math>n_i</math>);</li> </ol>
---

### 3.2 Scenario 2: single end node failure

End nodes are also referred to as leaf nodes. These are nodes normally present at the edge of the network. Failure of end-nodes does not affect inter-node connectivity. However, their failure affects the coverage area. Upon detecting end node failure (due to the absence of HELLO messages), the neighboring nodes will calculate the overlapped coverage area with the failed node using the mechanism presented in [19]. A neighboring node with more overlapping distance with the failed node is a suitable candidate for moving toward a failed node. During the movement, the node will continue to send HELLO messages and receive ACK messages to neighboring nodes with a communication range of  $R_c$ . This process ensures that the node mobility by a suitable candidate node does not cause network disconnectivity. For further illustration of this process, let us consider Fig. 3. Let us assume that node S4 has failed. The failure of S4 will be detected by its neighbors, i.e., S3, and S6 due to the absence of SYN messages from S4. Both the neighbors S3 and S6 will compute the relative overlapped sensing area with the failed node S4. Node S3 has more overlapped areas with the failed node than S6. Therefore, S3 will be selected as a suitable neighbor responsible for moving toward the failed node S4. According to [19], S3 will move a maximum distance of  $R_c/4$  toward the direction of S4. This process is illustrated in Fig. 4, and the steps are presented in Algorithm 3. During the movement, S3 will continue sending and receiving HELLO and ACK messages with  $R_c$ 's transmission range with neighbors to ensure connectivity. End nodes are also referred to as leaf nodes. These are the nodes normally present at the edge of the network. Failure of end-nodes does not affect the inter-node connectivity [25–28].





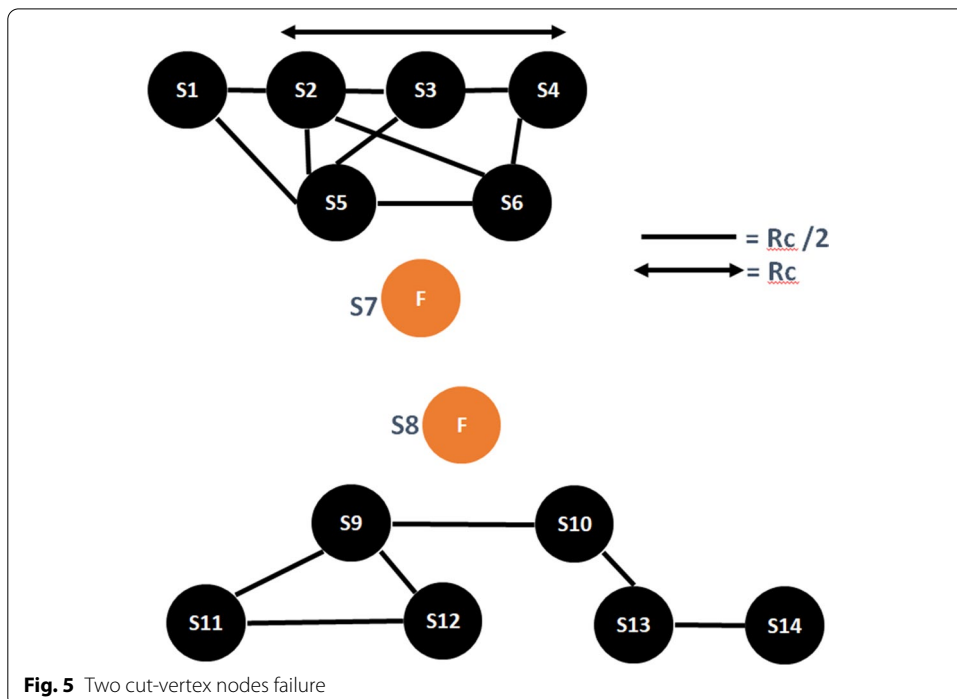
**Algorithm 3: SingleEndNodeFDR( $S, E, N, \gamma$ )**

**Input:**  $S$  ... Set of Sensor Nodes:  $\{s_i\}$   
**Input:**  $E$  ... Set of End Nodes:  $\{e_i\}$   
**Input:**  $N$  ... Set of Neighboring Nodes:  $\{n_i\}$   
**Input:**  $\gamma$  ... Scalar value as communication range  
**Output:** Single-end node failure successfully detected and restored

1. *foreach*  $e_i \in E$  **do**
2.      $n_i \leftarrow \text{Neighbour}(N, e_i)$                                  //end node's neighbour
3.     *if*  $n_i$  detects that  $e_i$  has failed **then**
4.          $A_i \leftarrow \text{Calculate\_Overlapped\_Area}(n_i)$  //neighbour of  $i$  position
5.          $A_j \leftarrow \text{Calculate\_Overlapped\_Area}(n_j)$  // neighbour of  $j$  position
6.         *if*  $A_i > A_j$  **then**
7.              $n_i$  moves towards  $n_j$  with a distance  $\gamma/4$  from its origin
8.             Broadcast HEARTBEAT message to all neighboring nodes
9.             Relocation();
10.            Update\_Routing\_Table( $s_i$ );

**3.3 Scenario 3: two cut-vertex node failure**

Two cut-vertex node failure is shown in Fig. 5, where multiple nodes fail simultaneously causing a network partition. The implications of two cut-vertex node failures are substantial. In this case, merely broadcasting HELLO messages with a transmission range of  $R_c$  will not be received by the nodes in the direction of a failed node as the distance between nodes is greater than  $R_c$  (as illustrated in Fig. 5). The absence of ACK will mean no immediate neighbors of the failed node in the  $R_c$  range. Dealing with this problem, a new type of message called coordination message will be broadcasted with a



**Fig. 5** Two cut-vertex nodes failure

transmission range of  $R_c/2$  among the neighbors as they start to move toward the failed node. The coordination message aims to ensure that no moving node goes out of range of its neighbor, causing further disruption in the network shows simultaneous node failure. These moving nodes will continue to transmit HELLO messages and wait for ACK messages. Receiving the ACK message will mean a node in the failed node’s vicinity capable of restoring connectivity. This process is illustrated in Fig. 6, and the steps are presented in Algorithm 4. Figure 6 shows the original position of the relocated nodes as well as the position after the relocation.

```

Algorithm 4: TwoCutVFDR( $S, C, N, \gamma$ )
Input:  $S$  ... Set of Sensor Nodes:  $\{s_i\}$ 
Input:  $C$  ... Set of Cut Vertex Nodes:  $\{c_i\}$ 
Input:  $N$  ... Set of Neighboring Nodes:  $\{n_i\}$ 
Input:  $\gamma$  ... Scalar value as communication range
Output: Two cut-vertex failure successfully detected and restored
1. foreach  $c_i, c_j \in C$  do
2.    $n_i \leftarrow \text{Neighbour}(N, c_i)$  //ith cut-vertex node’s neighbour
3.    $n_j \leftarrow \text{Neighbour}(N, c_j)$  //jth cut-vertex node’s neighbour
4.   if  $n_i$  detects that  $c_i$  has failed then
5.     if  $n_j$  detects that  $c_j$  has failed then
6.        $n_i, n_j$  move towards  $c_i$  and  $c_j$  with a distance  $\gamma/2$  to the origin
7.        $n_i, n_j$  broadcast HEARTBEAT with a range equals to  $\gamma$ 
8.       Cascaded_Movement();
9.       Update_Routing_Table( $s_i$ );
    
```

### 3.4 Scenario 4: multiple end node failure

Multiple end node failure is illustrated in Fig. 7. As WSNs operate in harsh environments, therefore multiple end node failure is a possibility. Multiple end node failure can

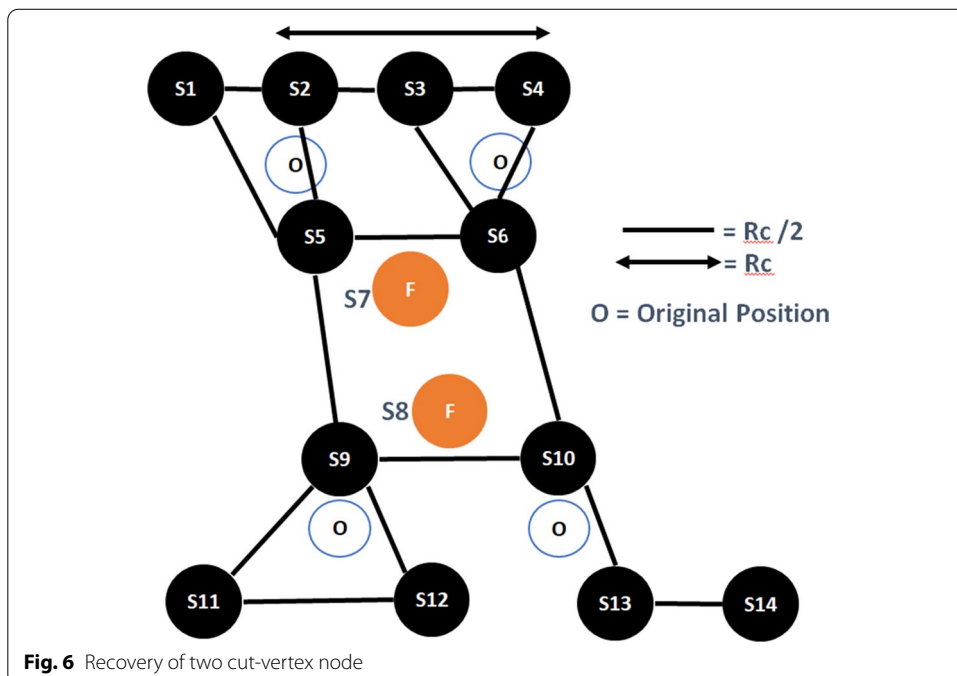
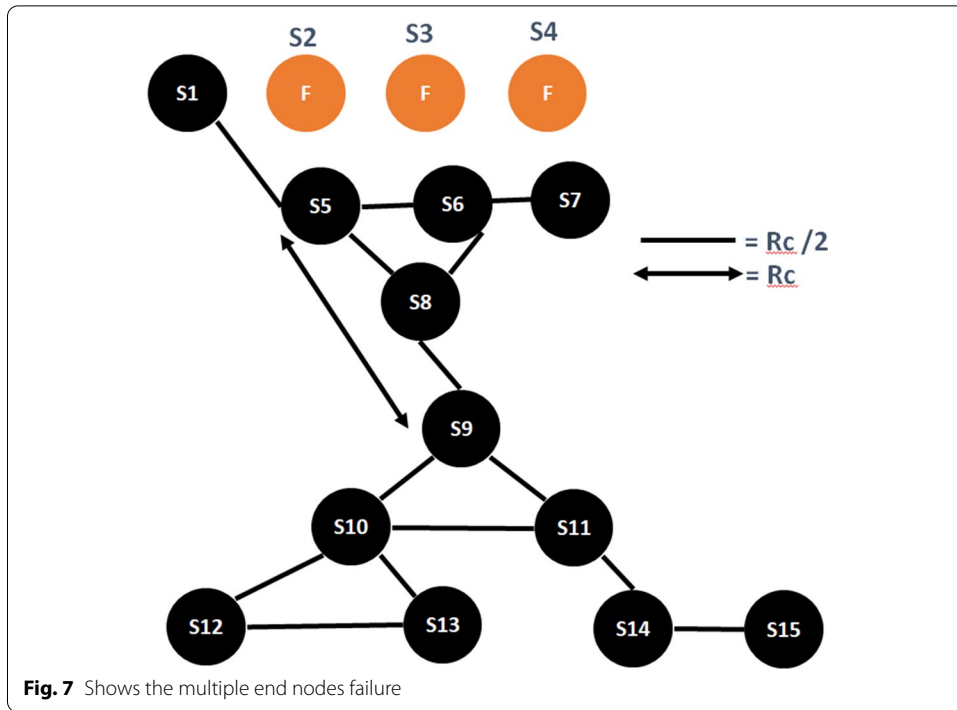


Fig. 6 Recovery of two cut-vertex node

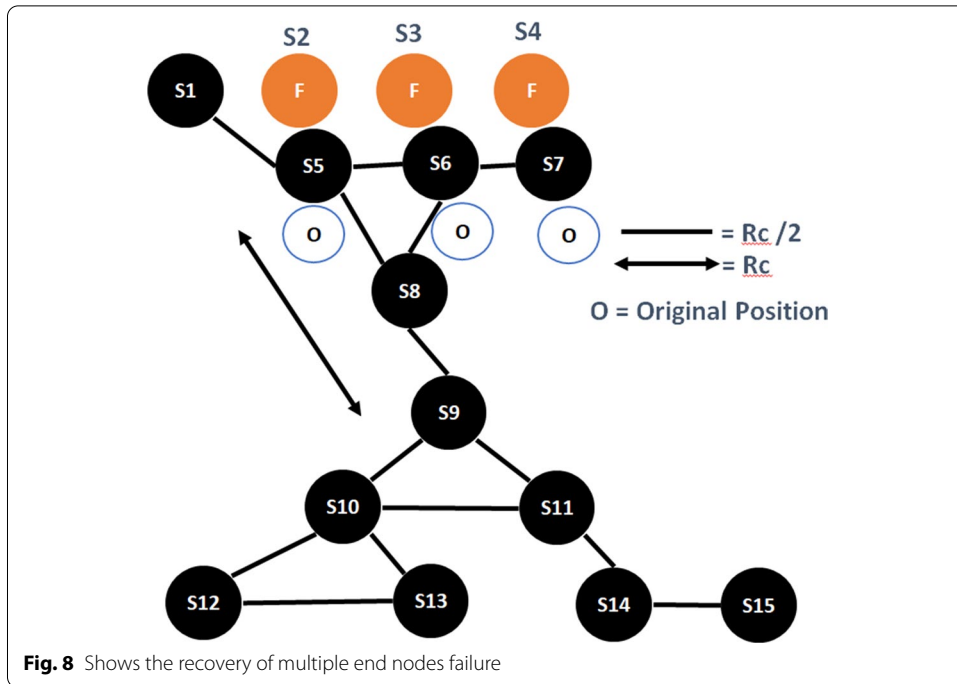


cause a big coverage hole, and for various applications, it is undesirable. To deal with such a situation, the failed nodes' neighbors start to move toward the failed nodes and exchange SYN messages for reporting the change in location to all neighbors. The maximum movement toward the failed nodes by neighbor nodes is  $Rc/4$  (as assumed in most baseline works such as [17–19]). Figure 8 illustrates the movement of nodes toward the failed nodes to cope with multiple node failures. Multiple end node failure detection and recovery process are presented in Algorithm 5.

<b>Algorithm 5:</b> MultiEndNodeFDR( $S, E, N, \gamma$ )	
<b>Input:</b> $S$ ... Set of Sensor Nodes: $\{s_i\}$	
<b>Input:</b> $E$ ... Set of End Nodes: $\{e_i\}$	
<b>Input:</b> $N$ ... Set of Neighboring Nodes: $\{n_i\}$	
<b>Input:</b> $\gamma$ ... Scalar value as communication range	
<b>Output:</b> Multi-end node failure successfully detected and restored	
1.	<i>foreach</i> $e_i \in E$ <b>do</b>
2.	$n_i \leftarrow \text{Neighbour}(N, e_i)$ //end node's neighbour
3.	<i>if</i> $n_i$ detects that $e_i$ has failed <b>then</b>
4.	$n_i$ moves towards $n_j$ with a distance $\gamma/4$ to its origin
5.	<b>Broadcast HEARTBEAT message to all neighboring nodes</b>
6.	<i>Relocation</i> ();
7.	<i>Update_Routing_Table</i> ( $s_i$ );

### 3.5 Algorithm analysis

There are five algorithms given above. Each of these algorithms takes a certain amount of time which is known as its complexity in terms of time denoted by  $T(n)$  where  $n$  is



**Table 1** Algorithm analysis

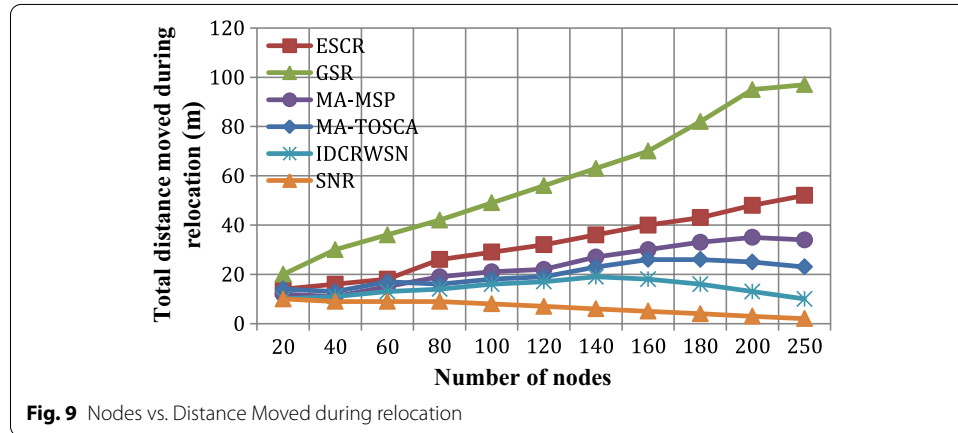
Step	Algo.1	Algo.2	Algo.3	Algo.4	Algo.5
1	$s_{i+1}$	$c_{i+1}$	$e_{i+1}$	$c_{i+1}$	$e_{i+1}$
2	$s_i$	$c_i$	$e_i$	$c_i$	$e_i$
3	$s_i$	$c_i$	$t_1: t_1 \leq e_i$	$c_i$	$e_i$
4	$t_1: t_1 \leq s_i$	$t_1: t_1 \leq c_i$	$t_1: t_1 \leq e_i$	$t_1: t_1 \leq c_i$	$t_1: t_1 \leq e_i$
5	$t_2: t_2 \leq t_1$	$t_1: t_1 \leq c_i$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_1: t_1 \leq e_i$
6	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_1: t_1 \leq e_i$
7	$t_2: t_2 \leq t_1$	$t_1: t_1 \leq c_i$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_1: t_1 \leq e_i$
8	$s_i * (n_i + 1)$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	-
9	$s_i * n_i$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	$t_2: t_2 \leq t_1$	-
10	$s_i * n_i$	-	-	-	-
11	$t_3: t_3 \leq s_i * n_i$	-	-	-	-
12	$t_4: t_4 \leq t_3$	-	-	-	-
13	$t_4: t_4 \leq t_3$	-	-	-	-
14	$t_4: t_4 \leq t_3$	-	-	-	-
$\Sigma$	$s_i(4 + 3n_i) + t_1 + 3t_2 + t_3 + 3t_{4+1}$	$3c_{i+1}3t_1 + 3t_{2+1}$	$2e_{i+1} 2t_1 + 5t_{2+1}$	$3c_{i+1} t_1 + 5t_{2+1}$	$3e_{i+1}4t_{1+1}$
Max $\Sigma$	$s_i(15 + 7n_i)_{+1}$	$9c_{i+1}$	$9e_{i+1}$	$9c_{i+1}$	$7e_{i+1}$
T(n)	$O( S )$	$O( C )$	$O( E )$	$O( C )$	$O( E )$

the input. The following table demonstrates the time complexity measurement of all five algorithms:

Table 1 gives the time cost of each step in terms of its input. For example, step 4 of Algo.1 is a while-statement under the loop. Its cost is  $t_1: t_1 \leq s_i$  which means it will execute  $t_1$  times depending upon condition but at most  $s_i$  times.  $\Sigma$  represents the sum of

**Table 2** Simulation parameters

Simulation parameters	Values
Simulation area	950 × 950m <sup>2</sup>
Number of nodes	20–250
Rc	25–150 m
Simulation tool	NS-2.34



**Fig. 9** Nodes vs. Distance Moved during relocation

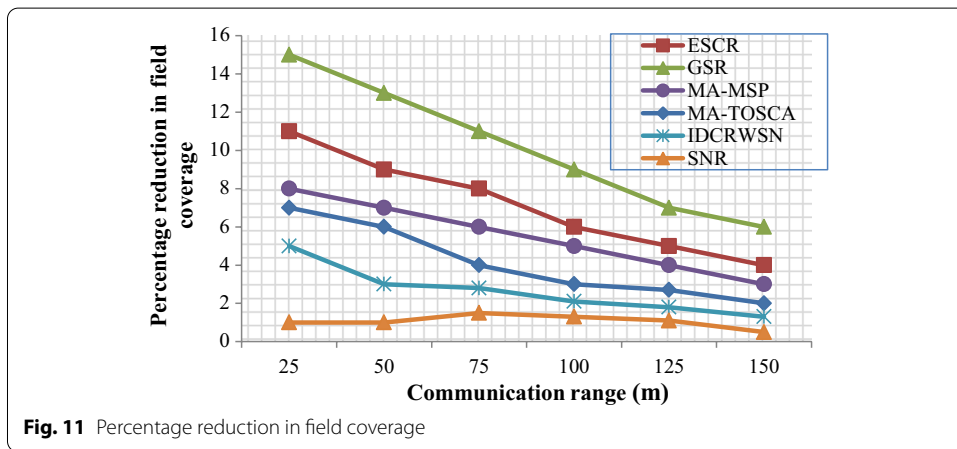
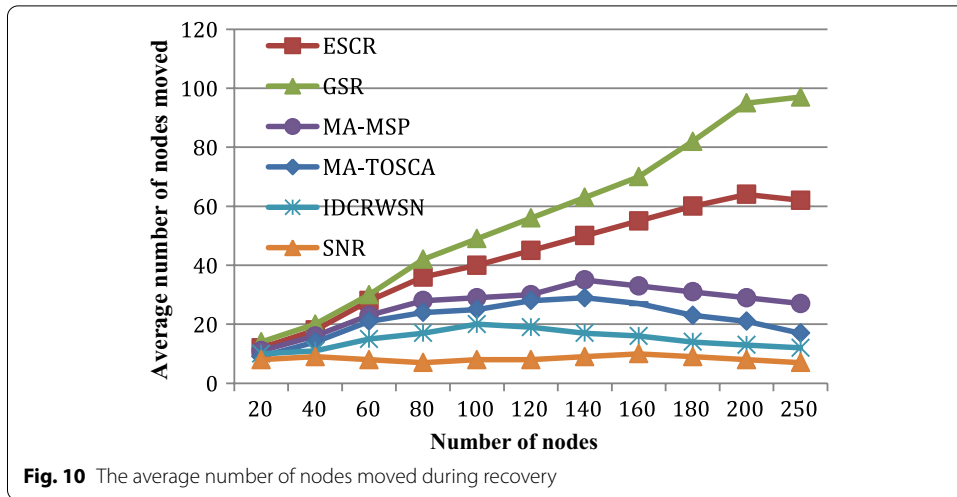
the cost of all steps involved in the respective algorithm.  $\text{Max}\Sigma$  means at most number of occurrences of  $t$  times. It can be easily seen that all algorithms are linear in their complexity depending upon their input size.

#### 4 Results and discussion

For the simulations, we have used NS2 (Network Simulator 2). During all simulations, at time  $T=0$ , sensor nodes are randomly deployed in a field with dimensions of  $950 \times 950 \text{ m}^2$ . The communication range is varied between 25 and 150 m. Node density is varied in the simulation area by varying the number of nodes between 20 and 250. Table 2 illustrates all the simulation parameters. Each point in the graph is calculated by running simulations with random seeds ten times. The results for the proposed algorithm are compared with baseline algorithms MA-TOSCA [18], MA-MSP [20], ESCR [21], GSR [22], and IDCRWSN [23]. The following sections explain the results obtained by doing extensive simulations.

##### 4.1 Total distance moved during relocation

Figure 9 shows the effect of increasing the number of nodes on the total distance moved by nodes for connectivity restoration. It can be observed that our proposed protocol SNR performs well compared to all the other baseline algorithms. The major reason behind this is that SNR moves just the critical nodes near the failed nodes. Alternatively, all the other baseline algorithms rely on non-critical nodes' movement, resulting in cascaded relocation. Therefore, the average distance moved by all baseline protocols is much more as compared to our proposed algorithm. Cascaded relocation results in an increase in the average distance moved by the nodes during recovery and average energy



consumption. As SNR reduces cascaded relocation compared to other protocols, therefore it proves to be more efficient.

#### 4.2 Number of nodes moved

The number of nodes moved by the considered protocols by increasing the total number of nodes in the network is presented in Fig. 10. As the number of nodes increases; the number of nodes moved by all protocols increases. However, our proposed protocol SNR outperforms all the considered baseline protocols as the increase in the number of nodes moved is lesser. Cascaded relocation is the main reason for more nodes moving on average for all the considered baseline algorithms. Cascaded relocation increases as the number of nodes in the network increase. It is evident from Fig. 10 that SNR is scalable and performs well as the number of nodes increases in the network.

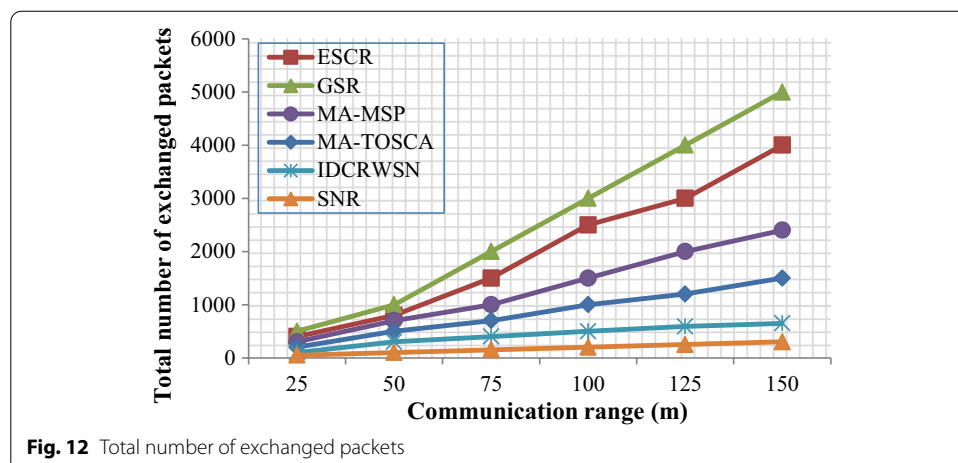
#### 4.3 Reduction in field coverage

The percentage reduction in field coverage concerning different communication ranges is shown in Fig. 11. Two factors contribute to the reduction in field coverage. First, the nodes that die due to complete drainage of their batteries; second, the node movement

to restore connectivity. Our proposed protocol aims to achieve connectivity restoration by reducing the number of exchanged messages (for achieving energy efficiency) and moving only critical nodes (for the restoration of connectivity and improving field coverage in case of failed nodes). It is evident from the figure that with an increase in the communication range of nodes, the percentage reduction in the field coverage decreases for all considered protocols. The percentage reduction in the field coverage by our proposed protocol SNR is lesser than all other baseline protocols. Among all considered protocols, GSR yields the largest percentage reduction in the field. The major reason behind this observation is using excessive cascaded relocation for connectivity restoration. Other protocols also move non-critical nodes for connectivity restoration, leading to the increased energy consumption of nodes due to movement leading to the failure of more nodes in the network. This ultimately leads to a decrease in the coverage area. For SNR, the percentage reduction in the field coverage remains below 2 percent for all the considered communication ranges.

#### 4.4 Number of exchanged packets

The average number of packets exchanged by all considered protocols is illustrated in Fig. 12. For the working of each protocol, several different types of packets are transmitted. For Fig. 12, the communication range is varied between 25 to 150 m. It can be observed from the figure that with the increase in the communication range, the number of packets exchanged increases for all considered protocols. The operational details of each protocol impact the number of packets that need to be exchanged. Moreover, the decisions made regarding the movement of nodes also play a key role. Whenever a node is relocated, it needs to exchange different control packets with its neighboring nodes. The more nodes a protocol relocates, the more packets are exchanged. It can be seen from the figure that GSR exchanges the maximum number of packets compared to all other protocols. Cascaded relocation is one of the major factors resulting in increased packets for all baseline protocols. As our protocol moves just the critical nodes; therefore, it avoids unnecessary relocation of nodes. This results in a lower number of exchanged packets. Due to this, our proposed protocol exchange the least number of packets. Exchanging the least number of packets also makes our proposed protocol





more energy efficient as packets' exchange requires energy. Therefore, SNR proves to be the most energy-efficient protocol among all considered protocols.

For the continuous operation of sensor networks, connectivity restoration is of immense importance, and a technique capable of restoring the connectivity is crucial for smooth operation. A connectivity restoration technique should be self-organizing, coverage-aware, and energy-efficient. By studying the literature, it was observed that most of the solutions for connectivity restoration focused on only one of the above features but not all of them collectively. This research aimed to design a novel connectivity restoration mechanism that effectively restores connectivity by moving fewer nodes than existing techniques. Another focus was to keep the connectivity restoration technique energy efficient by exchanging a minimal number of control messages. Last but not least, the technique should minimize the reduction in field coverage. Our proposed connectivity restoration mechanism achieves all the above objectives. Extensive simulations proved the effectiveness of our proposed protocol.

## 5 Conclusion

Node failures pose serious challenges for the researchers because they may lead to connectivity disruption among the nodes. It is inevitable to have an efficient connectivity restoration mechanism capable of efficiently restoring connectivity in case of single or multiple node failures. In this paper, a novel connectivity restoration technique called SNR is proposed. SNR is capable of detecting single and multiple node failures and efficiently restores connectivity by relying on the movement of the minimal number of nodes. It does not rely on the exchange of excessive control messages and also avoids the problem of cascaded relocation by relocating the minimal number of nodes for connectivity restoration. It also improves the field coverage as it results in a minimum percentage reduction in the field coverage compared to the other approaches.

Future work can be done in two possible directions. The first direction is the development of a simple yet flexible analytical model for calculating the performance metrics under different mobility models. The second direction involves the real-world implementation of the proposed solution for extensive performance evaluations.

## Abbreviations

WSNs: Wireless sensor networks; SNR: Smart node relocation; C<sup>2</sup>AP: Coverage-aware and connectivity-constrained actor positioning; COCOLA: Connected coverage and latency-aware actor placement; C<sup>3</sup>R: Coverage conscious connectivity restoration; DARA: Distributed actor recovery algorithm; GSR: Geometric skeleton based reconnection; IDCRWSN: Intelligent on-demand connectivity restoration for wireless sensor networks; PRACAR: Permanent relocation algorithm for centralized actor recovery; SRRA: Self-route recovery algorithm; ESCR: Efficient solution for connectivity restoration; ABN: Assigning backup nodes; CPR: Connectivity restoration process; ACK: Acknowledgment; Rc: Communication range; SYN: Synchronization; Nbr: Neighboring node.

## Acknowledgements

We would like to express our gratitude to King Khalid University for providing us administrative support and a healthy research environment.

## Authors' contributions

This work is a part of Dr. Mahmood ul Hassan' Ph.D. thesis work and he has contributed to all the sections of this work. Dr. Shahzad Ali as Dr. Mahmood's Ph.D. supervisor, he has worked on the state-of-the-art study and formalization of the problem statement. Dr. Khalid Mahmood and Dr. Muhammad Kashif Saeed actively assisted in the simulation setup and installation of the necessary software for carrying out research. Dr. Safdar Zaman has contributed to designing the algorithms and their complexities. Dr. Amin Al-Awady and Muhammad Saqib was involved in technical writing and grammar checking, and correction of the article. All authors read and approved the final manuscript.

### Availability of data and materials

The authors of this paper state that the simulator NS-2.34 generated the data used for performance evaluation purposes. The data can be accessed by using the simulator NS-2.34. Moreover, finally, there are no restrictions on data access as NS-2.34 is a free simulator. The simulation-based data used to support the findings of this study may be released upon application to Dr. Mahmood ul Hassan, who can be contacted at mahmood.mscs@gmail.com.

### Declarations

#### Competing interests

There is no financial and non-financial competing interest for this work.

#### Author details

<sup>1</sup>Department of Computer Skills, Deanship of Preparatory Year, Najran University, Najran, Kingdom of Saudi Arabia.

<sup>2</sup>Department of Information Systems, Tehama Branch, College of Science and Arts, King Khalid University, Abha, Kingdom of Saudi Arabia.

<sup>3</sup>Department of Computer Science, Tehama Branch, Community College, King Khalid University, Abha, Kingdom of Saudi Arabia.

<sup>4</sup>Department of Computer Science, Jouf University, Tabarjal Campus, Sakaka, Kingdom of Saudi Arabia.

<sup>5</sup>Department of Computer Science, Federal Directorate of Education, Islamabad, Pakistan.

<sup>6</sup>Department of Computer Science, Turaif Community College, Northern Borders University, Arar, Kingdom of Saudi Arabia.

Received: 18 February 2021 Accepted: 13 September 2021

Published online: 27 September 2021

### References

1. S. Lee, M. Younis, M. Lee, Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance. *Ad Hoc Netw.* **24**, 1–19 (2015). <https://doi.org/10.1016/j.adhoc.2014.07.012>
2. M.Z. Bhuiyan, G. Wang, J. Cao, J. Wu, Deploying wireless sensor networks with fault-tolerance for structural health monitoring. *IEEE Trans. Comput.* **64**(2), 382–395 (2013). <https://doi.org/10.1109/tc.2013.195>
3. M. ul Hassan, A. Al Awady, K. Mahmood, S. Ali, M.K. Saeed, Node relocation techniques for wireless sensor networks: a short survey. *Int. J. Adv. Comput. Sci. Appl. IJACSA* **10**(11), 323–329 (2019). <https://doi.org/10.14569/ijacsa.2019.0101145>
4. V. Ranga, M. Dave, A.K. Verma, Node stability aware energy efficient single node failure recovery approach for WSNs. *Malays. J. Comput. Sci.* **29**(2), 106–123 (2016). <https://doi.org/10.22452/mjcs.vol29no2.3>
5. M. ul Hassan, M.A. Khan, S. Ali, K. Mahmood, A.M. Shah, Distributed energy efficient node relocation algorithm (DEENR). *Int. J. Adv. Comput. Sci. Appl. IJACSA* **9**(3), 95–100 (2018). <https://doi.org/10.14569/ijacsa.2018.090315>
6. M.R. Senouci, A. Mellouk, L. Oukhellou, A. Aissani, WSNs deployment framework based on the theory of belief functions. *Comput. Netw.* **88**, 12–26 (2015). <https://doi.org/10.1016/j.comnet.2015.05.014>
7. B. Ishibashi, R. Boutaba, Topology and mobility considerations in mobile ad hoc networks. *Ad hoc Netw.* **3**(6), 762–776 (2005). <https://doi.org/10.1016/j.adhoc.2004.03.013>
8. A.A. Abbasi, M. Younis, A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.* **30**(14–15), 2826–2841 (2007). <https://doi.org/10.1016/j.comcom.2007.05.024>
9. K. Akkaya, M. Younis, Coverage and latency aware actor placement mechanisms in WSNs. *Int. J. Sens. Netw.* **3**(3), 152–164 (2008). <https://doi.org/10.1504/ijnsnet.2008.018476>
10. K. Akkaya, M. Younis, C2AP: Coverage-aware and connectivity-constrained actor positioning in wireless sensor and actor networks, in 2007 IEEE International Performance, Computing, and Communications Conference (2007), pp 281–288
11. M. Younis, S. Lee, A.A. Abbasi, A localized algorithm for restoring internode connectivity in networks of moveable sensors. *IEEE Trans. Comput.* **59**(12), 1669–1682 (2010). <https://doi.org/10.1109/tc.2010.174>
12. M. Younis, K. Akkaya, Strategies and techniques for node placement in wireless sensor networks: a survey. *Ad Hoc Netw.* **6**(4), 621–655 (2008). <https://doi.org/10.1016/j.adhoc.2007.05.003>
13. P. Basu, J. Redi, Movement control algorithms for realization of fault-tolerant ad hoc robot networks. *IEEE Netw.* **18**(4), 36–44 (2004). <https://doi.org/10.1109/mnet.2004.1316760>
14. N. Tamboli, M. Younis, Coverage-aware connectivity restoration in mobile sensor networks. *J. Netw. Comput. Appl.* **33**(4), 363–374 (2010). <https://doi.org/10.1016/j.jnca.2010.03.008>
15. M.K. Saeed, M. ul Hassan, A.M. Shah, K. Mahmood, Connectivity restoration techniques for wireless sensor and actor network (WSAN), a review. *Network* (2018). <https://doi.org/10.14569/ijacsa.2018.090919>
16. G. Wang, G. Cao, T.F. La Porta, Movement-assisted sensor deployment. *IEEE Trans. Mob. Comput.* **5**(6), 640–652 (2006). <https://doi.org/10.1109/TMC.2006.80>
17. A.A. Abbasi, K. Akkaya, M. Younis, A distributed connectivity restoration algorithm in wireless sensor and actor networks, in 32nd IEEE Conference on Local Computer Networks (LCN 2007) (2007), p. 496–503
18. X. Fu, P. Pace, G. Aloï, L. Yang, G. Fortino, Topology optimization against cascading failures on wireless sensor networks using a memetic algorithm. *Comput. Netw.* **177**, 107327 (2020)
19. U. Baroudi, M. Aldarwbi, M. Younis, Energy-aware connectivity restoration mechanism for cyber-physical systems of networked sensors and robots. *IEEE Syst. J.* **14**(3), 3093–3104 (2020)
20. X. Fu, H. Yao, Y. Yang, Modeling and optimizing the cascading robustness of multisink wireless sensor networks. *IEEE Trans. Reliab.* **70**(1), 121–133 (2021). <https://doi.org/10.1109/TR.2020.3024797>
21. M.K. Saeed, M. ul Hassan, K. Mahmood et al., Efficient solution for connectivity restoration (ESCR) in wireless sensor and actor-networks. *Wirel. Pers. Commun.* **117**(3), 2115–2134 (2020). <https://doi.org/10.1007/s11277-020-07962-3>

22. Y.K. Joshi, M. Younis, Exploiting skeletonization to restore connectivity in a wireless sensor network. *Comput. Commun.* **75**, 97–107 (2016). <https://doi.org/10.1016/j.comcom.2015.07.022>
23. K. Mahmood, M.A. Khan, A.M. Shah, S. Ali, M.K. Saeed, Intelligent on-demand connectivity restoration for wireless sensor networks. *Wirel. Commun. Mob. Comput.* (2018). <https://doi.org/10.1155/2018/9702650>
24. K. Mahmood, M.A. Khan, M. Ul-Hassan, A.M. Shah, M.K. Saeed, Permanent relocation and self-route recovery in wireless sensor and actor networks. *Int. J. Adv. Comput. Sci. Appl. IJACSA* **9**, 83–89 (2018). <https://doi.org/10.14569/IJACSA.2018.090313>
25. B. Chen, H. Chen, C. Wu, Obstacle-avoiding connectivity restoration based on quadrilateral Steiner tree in disjoint wireless sensor networks. *IEEE Access* **7**, 124116–124127 (2019). <https://doi.org/10.1109/ACCESS.2019.2938225>
26. L. Liu, M. Ma, C. Liu, W. Qu, G. Zhang, Y. Shu, ATCFS: effective connectivity restoration scheme for underwater acoustic sensor networks. *IEEE Access* **7**, 87704–87715 (2019). <https://doi.org/10.1109/ACCESS.2019.2921617>
27. Y. Zhang, Z. Zhang, B. Zhang, A novel hybrid optimization scheme on connectivity restoration processes for large scale industrial wireless sensor and actuator networks. *Processes* **7**(12), 939 (2019)
28. M. Hassan, K. Mahmood, S. Ali, A. Awady, M.K. Saeed, Node relocation techniques for wireless sensor networks: a short survey. *Int. J. Adv. Comput. Sci. Appl. IJACSA* **10**(11), 323–329 (2019)

### Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

---

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)

---