International Journal of
Mathematical Modelling & Computations
Vol. 03, No. 01, 2013, 71- 82

Energy Aware Distributed Partitioning Detection and Connectivity Restoration Algorithm in Wireless Sensor Networks

M. Jahanshahi [a*], M. Maddah [b] and N. Najafizadegan [c]

[a] *Department of Computer Engineering, Central Tehran Branch, Islamic Azad university, Tehran, Iran,*
[b] *Department of Computer Engineering, Qazvin Islamic Azad University, Qazvin, Iran,*
[c] *Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Qazvin, Iran.*

Received: 18 December 2012; Accepted: 11 March 2013.

Abstract. Mobile sensor networks rely heavily on inter-sensor connectivity for collection of data. Nodes in these networks monitor different regions of an area of interest and collectively present a global overview of some monitored activities or phenomena. A failure of a sensor leads to loss of connectivity and may cause partitioning of the network into disjoint segments. A number of approaches have been recently proposed that pursue node relocation in order to restore connectivity. DCR is a distributed partitioning detection and connectivity restoration algorithm to tolerate the failure of sensors. DCR proactively identifies sensors that are critical to the network connectivity based on local topological information, and designates appropriate, preferably non-critical, backup nodes. Upon failure detection, the backup sensor initiates a recovery process that may involve coordinated relocation of multiple sensors. Here we proposed Energy aware Distributed partitioning detection and connectivity restoration algorithm (EDCR) that is an improvement of DCR algorithm. Therefore reducing the message exchange overhead, lower energy consumption, and thus will increase the network lifetime.

Keywords: Wireless sensor networks, Connectivity restoration, Fault recovery, Node relocation, Enargy aware.

Index to information contained in this paper

* Corresponding author. Email: mjahanshahi@iauctb.ac.ir

1. Introduction

Interest in the applications of wireless sensor networks (WSNs) has been on the rise in recent years. For some of these applications, such as space exploration , coastal and border protection, combat field reconnaissance and search and rescue, it is envisioned that a set of mobile sensor will be employed to collaboratively monitor an area of interest, track certain events or phenomena and perform some tasks. By serving in such a harsh environment, WSNs can reduce cost and human risk. Nodes in these applications are typically empowered with limited processing and communication capabilities. Upon their deployment, they expected to form a network in order to share data and coordinate their action when participating in the execution of a task. In fact in many setups such as a disaster management application, nodes need to collaborate with each other in order to effectively search for survivors, assess damage and identify safe escape paths. To enable these interactions, nodes need to stay reachable to each other.

its clear that connectivity only requires that the location of any active node be within the communication range of one or more active nodes such that all active nodes can form a connected communication backbone. Onces ensor configuration is finished, nodes shall be comprised of connected networks to send information collected back to the control center [9].

Inter-node connectivity is not only very crucial to the effectiveness of the application, but some nodes may also play a role in maintaining flow of information from the sensors to in situ and remote users. The failure of a node could hamper the network connectivity and disrupt the collection of the sensed data. In the worst case, due to a node failure, the network may get partitioned into multiple disjoint blocks and stop functioning. Thus, the network connectivity should be recovered so that subsequent negative effects on the application could be avoided [7].

Rapid restoration of connectivity is desirable in order to maintain the WSN responsiveness to detected events. Deploying a replacement of the failed node is a slow solution at best and is often infeasible in risky areas, e.g., combat zones. Therefore, the recovery should be a self-healing process involving the existing nodes. Given the autonomous and unsupervised operation of WSN, tolerating the failure should be performed in a distributed manner. In addition, the overhead should be minimized in order to suit the resource-constrained sensors.

Most of the existing approaches in the literature are purely reactive with the recovery process initiated once the failure of F is detected. The main idea is replace the failed node F with one of its neighbors or move those neighbors inward to autonomously mend severed topology in the vicinity of F. Usually the repositioning of the neighbors of F causes more links to break and the relocation process repeats in a cascaded manner. Since these reactive schemes require coordination among the healthy nodes, the recovery process often imposes high messaging overhead.

In this paper, we present a reliable and energy centric distributed partitioning Detection and Connectivity Restoration (EDCR) algorithm, which proactively determines potential critical sensors and assign backup nodes in order to rapidly repair the topology with little overhead. First, each sensor proactively assesses its criticality, i.e., being a cut-vertex in the network topology, in a distributed manner based on the local information. Each critical (primary) sensor designates appropriate neighbor (preferably non-critical) as its backup. Once nodes onboard energy falls below a certain threshold, it informs the neighbor and its backup with sending a distress message. Neighbors and backup node continuously monitors primary for possible failure. Once the failure is detected, Neighbors buffer the data until done recovery and the backup initiates a recovery process by replacing the primary and when receives to primary s location sends Done Recovery message to neighbors that sends buffered data, so that the connectivity is restored. The algorithm is recursively executed until all nodes become strongly connected.

EDCR assumes single critical node failure at time, no other node fails during the recovery process, and the node that depleted energy set to failed node.

This paper is organized as follows. Section 2 discusses related works. The system model and problem statement is discussed in Section 3. The proposed EDCR algorithm is detailed in Section 4. Section 5 presents the analysis the improved recovery algorithm. Section 6 concludes the paper.

2. Related Works

The issue of fault tolerance in different WSN contexts has only been studied in few studies. The existing work on using node mobility to recovery from a failure can be categorized into block and cascaded movement. Block movement often requires a high pre-failure connectivity in order for the nodes to coordinate their response. For example, in some studies, the initial network is assumed to be 2-connected and goal is to sustain such 2-connectivity even under link or node failure.

In this paper, EDCR focuses on providing 1-connectivity. Block movements often becomes infeasible in absence of higher level of connectivity. Therefore, few researchers have pursued cascaded node movement or shifted relocation. The idea is to gradually replace intermediate nodes on the path instead of moving a node for a long distance.

Strategies adopting cascaded relocations can be further categorized based on the network state information that nodes are assumed to maintain. Some approaches like DARA[1], require each node to maintain 2-hop neighbors. Others, such as RIM [8], C^{3}R[7], DCR[3] avoid the increased overhead for tracking 2-hop neighbors and require each node to maintain only its directly reachable nodes, i.e. 1-hop neighbors. Like our improved EDCR algorithm, DARA [4] strives to restore connectivity lost due to failure of cut- vertex. However, DARA requires more network state in order to ensure convergence. Meanwhile, in PADRA [2] identify a connected dominating set (CDS) of the whole network in order to detect cut-vertices. Since the CDS based method is not accurate for critical node detection, they perform a depth-first search (DFS) on each member for the CDS to confirm that the node is really a cut vertex or not. Although, they use a distributed algorithm, their solution still requires 2-hop neighbors information that increases messaging overhead.

Although RIM [8] , C^{3}R [7] and VCR [4] use 1-hop neighbor information to restore connectivity, they are purely reactive and do not differentiate between critical and non-critical nodes. Whereas, DCR is a hybrid algorithm that proactively identifies critical nodes and designates for them appropriate backups but it does not considered energy as a factor in designate backup node.

RAM [3],CoMN2 [5] design to handle one possible case of a multi-node failure although CoMN2 is a Centralized algorithm and consider the presence of specific entity called mobile node which has the ability of processing data and making the appropriate decision. In this approach, mobile node is a single point of failure.

Another work that proposed in [3] DCR uses a localized scheme to identify critical sensors and designate backups for them. The backup sensor detects the failure of the primary and pursues node relocation to repair the partitioned network topology. DCR considers one failure at a time and no other node fails during the recovery but it does not considered energy as a factor in designate backup node and also may lost some critical sensed data during relocation primary with backup node. EDCR propose here to improved DCR performance.

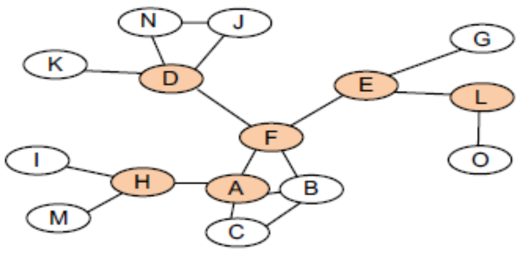

Figure 1. An example of connected inter-sensor network topology.

3. System Model and Problem Statement

The communication range (r_c) of a sensor refers to the maximum Euclidean distance that its radio can reach. To simplify analysis, nodes are assumed to have same communication range. Sensors are deployed randomly in an area of interest. We assume that a sensor can determine its location using an onboard GPS receiver, or position relative to its neighbors using localization techniques . Each node maintains a list of direct (1-hop) neighbors.

The impact of a nodes failure depends on the position of that sensor in the network topology. A node is said to be critical, cut- vertex in graph theory terminology, if its removal partitions the network into disjoint segments. The failure of one or multiple critical nodes not only affects the sensor coverage but significantly, impacts inter sensor connectivity. For example, consider a network topology depicted in Figure 1 Losing a leaf/non-critical node, such as G does not affect inter-sensor connectivity. Meanwhile, the failure of a critical node such as F partitions the network into disjoint blocks. This paper focuses on restoring inter-sensor connectivity lost due to failure of one adjacent critical sensor.

In order to tolerate critical node failure, three methodologies can be identified :(i) proactive,(ii) reactive and (iii) hybrid. Proactive approaches establish and maintain bi-connected topology in order to provide fault tolerance. This necessitates large sensor count that leads to higher cost and becomes impractical. On the other hand, in reactive approaches the network responds only when a failure occurs. Therefore, reactive approaches might not be suitable for time critical applications. In hybrid approaches, pre-failure planning is pursued in order to increase the efficiency of the recovery. We argue that a hybrid approach better suits autonomous WSNs that are deployed for time-critical applications due to the reduced recovery time and overhead. overhead [3].

4. EDCR Algorithm

In this paper, we propose EDCR algorithm. Since the most of failure in WSN is due to energy depletion, here we assume failure as depletion of sensor on board energy that is an improvement of DCR algorithm.

EDCR algorithm is hybrid in the sense it consists of two parts, i.e. proactive and reactive. The only pre-failure knowledge that EDCR requires for each node is to have is a list of 1-hop neighbors. This list is formed post deployment with each node broadcasting a HELLO message to introduce itself to its neighbors. Each node also tabulates the position and ID of all its neighbors. In the proactive part, critical nodes are determined using a localized algorithm. Once critical nodes (primary) are determined, they select and designate an appropriate neighbor (backup) to

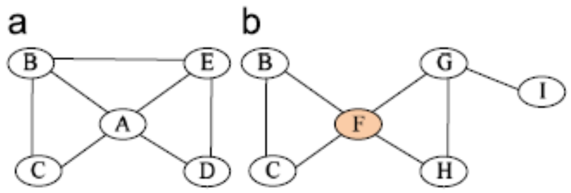

Figure 2. A segment of showing 1-hop positional:(a)critical and (b)non-critical sensors

handle their failure when such contingency arises in the future. In the reactive part, once nodes onboard energy falls below a certain threshold informs its 1-hop neighbors also the backup through send a DISTRESSMESSAGE. Neighbor and backup node start monitoring the primary through HEARTBEATS. A backup initiates a recovery process when the primary fails. Once neighbor nodes detect the failure they buffered critical data for prevent loss sensed data. The backup replaces the primary and informs new neighbor through send DONERECOVERY message. Neighbor nodes resumes routing of their buffered data packets with receive DONERECOVERY message.

4.1 *Identifying Critical Sensors*

Several algorithms to identify cut-vertices in a graph, critical nodes in the context of WSNs, have been proposed in the literature. These algorithms can be categorized into centralized and distributed. Centralized algorithms require each node to be aware of global topology. These methods involve huge communication overhead due to the dynamic nature of these networks. These methods involve huge communication overhead due to the dynamic nature of these networks. Distributed detection algorithms [4] , [10] are based on CDS and requires 2-hop neighbor information. Some localized algorithms require only 1-hop neighbors positional information at the expense of lower accuracy of cut-vertices identification. Basically, some nodes are marked as critical while they are not cut-vertices. However, no critical node will be missed.

Each node determines locally whether it is critical or not based on neighbors position information. It calculates the distance between neighbors based on their positions. If the distance is less than their communication range, the node is considered non- critical because neighbors would stay connected without it [3]. On the other hand, if the 1-hop neighbors of a node can be partitioned into more than one segment, the node is 1-hop critical. For instance, Figure 2 shows a localized scope of non-critical node A and critical node F. Nodes B, C, D, and E are 1-hop neighbors of node A as shown in Figure 2(a). Node A is 1-hop positional non-critical because its neighbors remain connected without A. On the other hand, neighbors of F can be divided into two sub graphs i.e.B, C andG, H, I. Therefore, F is 1-hop positional critical as illustrated in Figure 2(b). Furthermore, leaf nodes such as I, are detected as non-critical.

4.2 *Backup Selection*

Once the critical nodes (primary) are identified, the next step is to select and designate appropriate neighbors as backups. The purpose of the pre-nomination of backup nodes is to instantaneously react to the failure of critical nodes and avoid the possible network partitioning caused by such a failure. The nodes maintain minimum state information (i.e. 1-hop neighbors) to avoid extra overhead of messaging.

Since, neighbors become disconnected when a critical sensor fails, backup nodes are determined and notified before a failure of critical nodes takes place. The selection of a backup among 1-hop neighbors is based on the following ordered criteria:

(a) Neighbor sensor status (NSS): As discussed above, each sensor determines whether it is critical or non-critical. A non-critical neighbor sensor is preferred to serve as backup. This will limit the scope of the recovery , reduce incurred overhead and minimize the impact on coverage.
(b) Sensor degree (SD): A non-critical neighboring node (preferably leaf) is a more suitable candidate for backup since moving that node will have minimum impact on inter-sensor connectivity. If a non-critical node is not available in the neighborhood, EDCR prefers to choose a strongly connected critical node (with high degree) because there is more probability to have non-critical nodes in the neighborhood. This will limit the scope of cascaded relocation and thus lower the recovery overhead.
(c) Inter-sensor distance (ID): A close backup node is preferred in order to reduce the movement overhead. Again, it has to be feasible for the backup to travel to the position of the primary per the first criterion above.
(d) Remain Energy (RE): A neighbor node with the most remains energy is preferred for backup node, because failed later. To calculate consumption energy used energy model in [6].

$$(k,d) = \begin{cases} k(E_{\text{elec}} + \varepsilon_{\text{FS}})\ d < d_0 \\ k(E_{\text{elec}} + \varepsilon_{\text{MP}})\ d > d_0 \end{cases} \tag{1}$$

Where the energy dissipation (k,d) of transmitting k-bit data between two nodes separated by a distance of d meters and $d_0 = \sqrt{(\varepsilon_{\text{FS}}/\varepsilon_{\text{MP}})}$ and is quit E_{elec}.denotes electronic energy, ε_{FS} and ε_{MP} denote transmit amplifier parameters corresponding to the free-space and the two-ray models. The energy dissipation incurred in the receiver of the destination sensor node is

$$E_R(k) = k * E_{\text{elec}} \tag{2}$$

Also, the energy dissipation of fusing k-bits data is

$$E_F(k) = k * E_{\text{df}} \tag{3}$$

The parameters used in this report are given below: $E_{df} = 5nJ/bit, \varepsilon_{\text{FS}} = 10pJ/bit/m^2, E_{elec} = 50nJ/bit, \varepsilon_{MP} = 0.004pJ/bit/m^4$. The total energy calculated as fallow

$$E_{total} = E_T(k,d) + E_R(k) + E_{cpu}(k) + E_{mov} \tag{4}$$

A node may be selected as a backup for more than one sensor. In case a backup node fails or moves outside the range of its primary. Since the set of candidate backups is limited to the 1-hop neighbors, the picked backup may not be globally

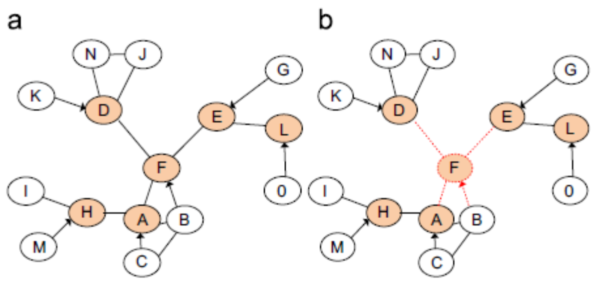

Figure 3. Critical sensors designate their backup using EDCR for network segment shown in Fig. 2: (a) backups start monitoring their primary and (b) B detect failure of primary F.

optimal. Nonetheless, the local selection enables EDCR to be applied in a distributed manner and scale for large networks. Figure 3(a) shows the setup where critical sensors appoint their backups. The arrow head point towards the primary. Note that EDCR does not require extra nodes for serving as backup. It employs existing sensors just to take care of each other.

4.3 *Failure Detection*

Once nodes onboard energy falls below a certain threshold informs its 1-hop neighbors also the backup through send a DISTRESS MESSAGE. Neighbor and backup node start monitoring the primary through HEARTBEATS. Figure 3 (b) indicates that the backup node B detects the failure of primary F and triggers the recovery process as detailed in the following section.

4.4 *Recovery Process*

The scope of the recovery depends on the NSS. If the backup is a non-critical, it simply replaces the primary and the recovery would be complete. However, if the backup is also critical node, cascaded relocation is performed. Basically, repositioning of node A_i in response to the failure of A_f will be interpreted by its backup A_j as if A_i is lost and A_j will thus move to replace A_i. The recovery process consists of following steps.

4.4.1 Primary Recovery

The backup node immediately initiates a recovery process once it detects failure of its primary. The scope of recovery depends on the position of backup node, which can be one of the following three scenarios.

1) If a backup is a non-critical node the scope of the recovery will be limited because it does not require further relocations. The backup node moves to the position of the failed primary. It selects and designates a new backup since it has become a critical node at the new position. This movement alerts the other primary nodes (if any) at the previous location to choose a new backup for themselves. An illustrative example is provided in Fig. 4, where non-critical backup B simply replaces its primary (i.e. F) and selects a backup for itself.
2) The second scenario is when the backup is also a critical node. In this case, the backup node will notify its own backup so that the network stays connected. This scenario may trigger a series of cascaded repositioning of nodes as explained below.
3) The third scenario is when the failed (primary) and its backup are both

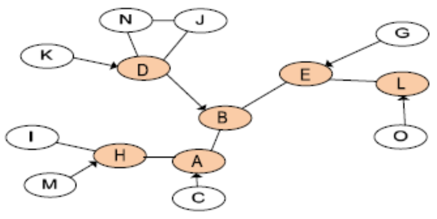

Figure 4. Recovery process when backup sensor non-critical.

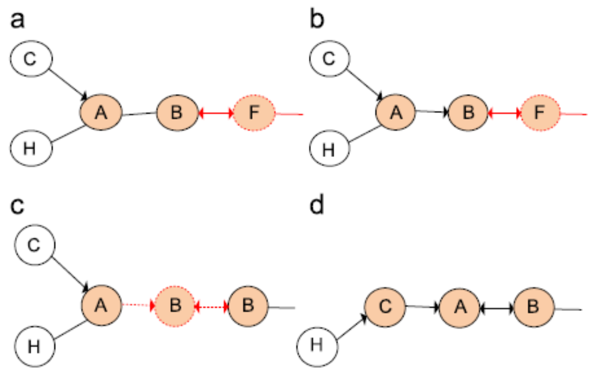

Figure 5. Applying the recovery process when two nodes are simultaneously primary-backup of each other.

critical nodes and simultaneously serving as backup for each other. This scenario is articulated in Figure 5. Node B detects the failure of F as both are mutually serving as backup for each other as shown in Figure 5(a). Figure 5(b) shows that the node B selects another node A as backup. Then B sends a movement notification message and moves to the position of F as shown in Figure 5(c). This movement triggers a series of cascaded relocations as discussed below and is shown in Figure 5(d), with A replacing B and C replacing A.

4.4.2 Cascaded Relocation

As mentioned earlier, the position of that backup determines the scope of the recovery. In particular, the recovery process of the second scenario is repeated to handle the departure of a backup node. Before the critical backup move to failed node location, notify its neighbors through sending a message. This process may be again applied by backup node and so on until a non-critical backup replaces a primary. Figure 6(a) illustrates this scenario where the backup node is also critical and the recovery process continues in a cascaded manner. Figure 6(b) shows restored Network.

The pseudo code of the EDCR algorithm is presented in algorithm 1.

Algorithm 1

```
EDCR(A)
1. if(Is-Critical(A)==True) Then
2. AssignBackup(A)
3. If(A energy goes under the critical-level) Then
4. Broadcast DistressMessage
5. End if
```

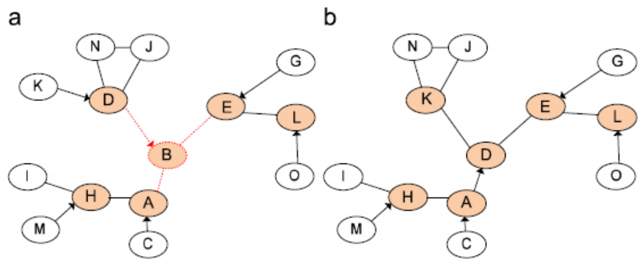

Figure 6. Illustrating the recovery process when backup node is critical:(a)the critical node D detects failure of primary B and (b) D replaces Band K replace D.

```
6. End if
7. If(A receive DistressMessage) Then
8. Failure Detection(A,F)
9. End if
10. If(A received Recovered Message) Then
11. Send Buffered Data

  Failure detection(A,F)
12. If(A detects missing Heartbeat Message from F) Then
13. If(A->BackupStatus()==Failed) Then
14. Buffer Data
15. Else
16. Recover(A,F)
17. End
18. End if

  Recover (A,F)
19. If(Is-Critical(A)==True) Then
20. If(A->BackupStatus()==Failed) Then
21. AssignBackup(A)
22. End if
23. NotifyBackup(A)
24. End if
25. Move to location (A,F)
26. Notify New Neighbors with send Recovered Message

  AssignBackup(A)
27. //Assign noncritical neighbor with highest degree and least distance node as
Backup
```

5. Performance Evaluation of EDCR

In this session we compare EDCR algorithm with DCR in 3 factor, consume energy, total distance moved and the number of exchanged messages.

$Numberofexchangedmessages$: In EDCR algorithm 6 type of messages exchanged between sensors. These consist of deployment message for deploy in the field, Distress message when a sensors energy goes under the threshold, Done Recovery message to notify neighbor nodes that the recovery process and send buffered data, Notify message if the backup is critical and Heartbeat message from receive

Distress message to failure detection.

$$N_{total(EDCR)} = N_{DistM} + N_{DoneR} + N_{dep} + N_{ABu} + N_{NBu} + N_{HB} \quad (5)$$

$N_{DistM}(DistressMessage)$: Once nodes onboard energy falls below a certain threshold, it informs the neighbor and its backup with sending a distress message. In the worst case the number of these messages is (n-1).

$N_{DoneR}(DoneRcovery)$: Once backup node receives to primary s location sends Done Recovery message to neighbors that sends buffered data. In the worst case, the number of these is (n-1). Ndep (Deployment): The Number of message that sensors exchange between each other when deployed in the field.

$N_{ABu}(AsignBackup)$: Nodes after assign backup, notify this through sending a message.

$N_{NBu}(NotifyBackup)$: If the backup node is critical, before the critical backup move to failed node location, notify its neighbors through sending a message. The number of these messages will vary depending on the network topology and in the worst case is (n-2).

$N_{HB}(HeratBeat)$:

$$N_{HB(EDCR)} = \Delta t/\boldsymbol{\tau} * Ngh \quad (6)$$

Where Ngh is the average number of neighbors per node and is periodic time that exchange HB messages.

$$\Delta t = |t_{alert} - t_{detect}|$$

$$N_{DistM} = N_{DoneR} = Ngh$$

$$N_{(EDCR)} = N_{DistM} + N_{DoneR}$$

$$N_{(EDCR)} = \Delta t/\boldsymbol{\tau} * Ngh + 2Ngh \quad (7)$$

If we consider as failure rate $e^{-\lambda t}$ is probability of failure so total of exchanged message is obtained as fallow

$$N_{total(EDCR)} = e^{-\lambda t}(\Delta t/\boldsymbol{\tau} * Ngh) + N_{similiar} \quad (8)$$

where $N_{similiar}$ is

$$N_{similiar} = N_{dep} + N_{ABu} + N_{NBu}$$

DCR algorithm has 4 diffrent type of message for exchange between sensors.

$$N_{total(DCR)} = N_{dep} + N_{ABu} + N_{NBu} + N_{HB} \quad (9)$$

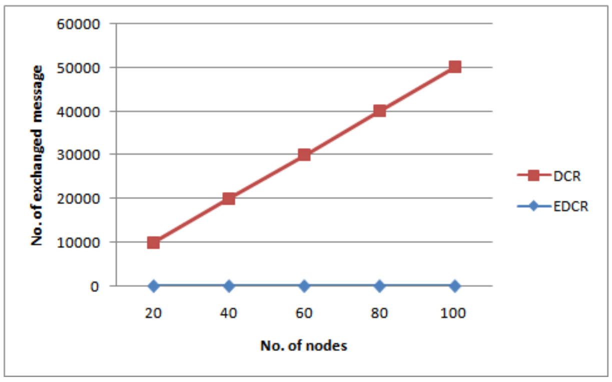

Figure 7. Illustrating the recovery process when backup node is critical:(a)the critical node D detects failure of primary B and (b) D replaces Band K replace D.

After deployment, sensors are assumed to discover each other and start exchange heartbeat messages with 1-hop neighbors to failure detection.

$$N_{HB(DCR)} = T/\boldsymbol{\tau} * n \tag{10}$$

Where T is network lifetime and n is total nodes. With Equ. (9) and (10) since the T is much larger than Δt, so heartbeat overhead decreased. In a result, messaging overhead in EDCR is much lower than DCR.

$$N_{total(DCR)} = (T/\boldsymbol{\tau}) * n + N_{similiar} \tag{11}$$

$$T >> \Delta t \Longrightarrow N_{total(DCR)} > N_{total(EDCR)} \tag{12}$$

Energy consumption: According to Equ.8 since In EDCR the messaging exchange overhead is lower than DCR so energy consumption decrease and consequently will increase the network lifetime. Total distance movement: energy depletion due to the physical movement of nodes is an obvious concern. Since EDCR method consider sensor energy as a measure for backup selection so the backup node failed later so total distance movement will decrease.

Figure 8. Indicates that the performance of EDCR in terms of the number of HB exchanged message is much better than DCR. as the figure shows, with increase nodes in network, number of HB message increase linear and figure 8. show that the number of exchanged message Significantly reduced.

6. Conclusion

This paper has presented an improvement distributed hybrid movement control algorithm for restoring connectivity lost due to critical sensor failure. The proposed EDCR algorithm identifies critical sensors in advance based on localized information and designates for them backup nodes . EDCR pursues controlled node relocation in order to reorganize the topology and regain the pre-failure strong connectivity. This method consider sensors energy as a measure to designate backup node so the node failed later and the number of exchange messages will decrease.

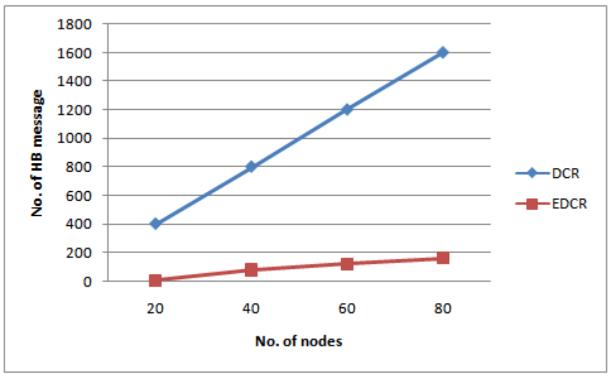

Figure 8. effect of changing n in total number of exchanged message.

In EDCR the node that depleted energy set to failed node and assume a threshold for each sensor energy. Once nodes onboard energy falls below a certain threshold, it informs the neighbor and its backup with sending a message. Therefore, in compare with DCR decrease messaging overhead and consuming energy and increase the Network lifetime. When primary node is going to failure location neighbors buffer the data until done recovery so EDCR ensures that no data lost. In the future, we plan to evaluate the performance of the proposed approaches in a prototype network of mobile robots and improve the cascade relocations will investigate.

References

[1] Abbasi A., Akkaya K. , Younis M., A Distributed Connectivity Restoration Algorithm in Wireless Sensor and Actor Networks, Proc.of 32nd Conf. on Local Computer Networks, Dublin, Oct.(2007) 496-503.

[2] Akkaya, K, Thimmapuram, A, Senel, F, Uludag, S., Distributed recovery of actor failures in wireless sensor and actor networks Proceedings of the IEEE wireless communications and networking conference (WCNC 2008), Las Vegas, NV; March (2008).

[3] Imran M. , Younis M. , Said A.M., Hasbullah H., Localized motion-based connectivity restoration algorithms for wireless sensor and actor networks, Journal of Network and Computer Applications, vol.**35** doi:10.1016/j.jnca.(2012) 844-856.

[4] Imran, M, Younis, M, Said, AM, Hasbullah, H., Volunteer-instigated connectivity restoration algorithm for wireless sensor and actor networks, Proceedings of the IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS 2010), Beijing, China; June (2010).

[5] Maalel N., Kellil M., Roux P., Bouabdallah A., Fast Restoration of Connectivity for Wireless Sensor Networks, Internet Of Things, Smat Space And Next Generation Networking Lecture Notes in Computer Science, **7469**, $doi : 10.1007/978 - 3 - 642 - 32686837$, Springer (2012) 401-412.

[6] Qiu M. , Liu J , Li J., Fei Z., Ming Z., Edwin H., Sha M. A Novel Energy-Aware Fault Tolerance Mechanism for Wireless Sensor Networks, IEEE/ACM International Conference on Green Computing and Communications ,4-5 Aug. 2011,pp.56-61,doi:10.1109/GreenCom (2011).

[7] Tamboli. N, Younis. M, Coverage-aware connectivity restoration in mobile sensor networks, Journal of Network and Computer Applications , **33**, doi:10.1016/j.jnca.(2010) 363-374 .

[8] Younis M., et. al. A Localized Self-healing Algorithm for Networks of Moveable Sensor Nodes, Proc. of IEEE Global Telecommunications Conf. (Globecom08), New Orleans, LA,, November. (2008) 1-5 .

[9] Zhu C., Zheng C., Shu. L, Hang. G, A survey on coverage and connectivity issues in wireless sensor networks, Journal of Network and Computer Applications, **35** 619-632. doi: 10.1016/j.jnca.2012.

[10] Zamanifar, A., Sharifi, M., Kashefi, O., A Hybrid Approach to Actor-Actor Connectivity Restoration in Wireless Sensor and Actor Networks , Networks , ICN'09. Eighth International Conference on IEEE (2009) 76-81.