Partition-aware centrality measures for connectivity restoration in mobile sensor networks

Izzet Fatih Senturk

Department of Computer Engineering,
Bursa Technical University,
Bursa, 16310, Turkey
Email: izzet.senturk@btu.edu.tr

Abstract: Mobile sensor networks (MSNs) often operate unattended in environments where human intervention is limited. To sustain network operations, network connectivity must be maintained at all times. However, the network can be partitioned due to random node failures. To tolerate such failures in a reactive manner, network topology can be restructured through node mobility. Minimising the mobility cost requires addressing two different challenges. First, identifying nodes to be relocated. Second, determining target locations for movement. We address the first problem by presenting three different partition-aware centrality measures based on closeness centrality, geometric centrality, and harmonic centrality. To determine the movement target, we consider the former locations of the upstream nodes so that simultaneous node failures can be tolerated with limited data collection scope. The approaches that we present in this paper not only ensure recovery but also minimise the recovery cost so that the network lifetime is extended.

Keywords: Mobile sensor networks; topology management; closeness centrality; geometric centrality; harmonic centrality; connectivity restoration; fault tolerance; mobility.

Reference to this paper should be made as follows: Senturk, I.F. (xxxx) 'Partition-aware centrality measures for connectivity restoration in mobile sensor networks', *Int. J. Sensor Networks*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Izzet Fatih Senturk earned his Bachelor's degree in Computer Engineering from Ege University in 2006 and Master's and PhD degrees in Computer Science from Cornell University and Southern Illinois University-Carbondale in 2008 and 2013, respectively. He is currently an Assistant Professor in the Department of Computer Engineering at Bursa Technical University. He is a Member of Institute of Electrical and Electronics Engineers. His research interests include wireless sensor networks, mobile computing, pervasive computing, and cloud computing.

1 Introduction

Mobile sensor networks (MSNs) consist of spatially distributed autonomous sensor nodes to monitor the surrounding physical phenomena such as temperature, humidity, pollution, sound, pressure, etc. Mobility can be employed either randomly (Vecchio et al., 2010) or in a controlled manner (Natalizio and Loscrí, 2013). Due to their limited size and the cost concerns, nodes are subject to severe resource constraints in terms of energy and computational power. Limited on-board power supply of the nodes imposes restrictions on the communication range. To connect the network to the end user, one or more Base Stations (BS) are employed. BS is typically resource-rich and acts as a gateway between the sensor nodes and the remote server. Due to the limited communication range of the sensor nodes, it is essential to coordinate their actions and collaborate in order to communicate with the BS. Connectivity with the BS can be sustained through multi-hop routing on the formed ad-hoc network so that the sensor readings can be delivered cooperatively.

Applications such as battlefield surveillance, forest fire detection, volcano or glacier monitoring and landslide detection, etc. expose nodes to harsh environmental conditions. Such physical surroundings render the nodes susceptible to arbitrary external damage. Battery depletion and random hardware malfunction also inflict the sensor nodes incurring permanent node failures. In such a case, redundancy may alleviate the impact of the failures if nodes can find an alternative path to reach the BS. However, an alternative path may not always be available. The node which serves on a path exclusively is regarded as a cut-vertex node. Removal of cut-vertex node(s) from the network partitions the network into multiple disjoint segments isolated from the rest of the network. In such cases, mobility can be exploited to restore the network connectivity. Restoring inter-segment connectivity is

essential so that the MSN becomes operational again. To avoid risking human life and expedite the recovery, it is desirable to pursue an unattended self-healing approach.

Mobility-based connectivity restoration schemes have been applied in response to the loss of single (Abbasi et al., 2007; Akkaya et al., 2008; Younis et al., 2010) or multiple (Sir et al., 2011; Alfadhly et al., 2010; Senturk et al., 2012a; Akkaya et al., 2013; Senturk et al., 2012b; Senturk and Akkaya, 2012, 2014) nodes in the MSNs. The simultaneous failure of multiple collocated nodes is more challenging compared to the single node failures both in analysing the scope of the failure and also providing a recovery solution (Younis et al., 2014a). The common approach is restructuring the network topology by exploiting mobility while minimising the mobility cost. The fundamental issues of this process are identifying the node (i.e., leader) to be relocated and determining the target location for movement. If the movement of a single node is not sufficient to restore connectivity, cascaded movement is pursued where a new leader node is selected in the successive steps to sustain recovery. To identify the leader node, various metrics can be considered such as node degrees (Abbasi et al., 2007), connectivity (Akkaya et al., 2008), or centrality (Senturk and Akkaya, 2014; Senturk, 2017).

In this paper, we present three different approaches based on centrality measures which are commonly used in social network analysis to identify key persons within the network. In our case, key person denotes central nodes in the control of information flow and the connectedness of the network. We present partition-aware closeness, geometric, and harmonic centrality measures to assess the importance of the nodes and select the node with the least centrality score for movement. For a given node, closeness centrality considers the length of the shortest paths to all other nodes in the network and appreciates nodes requiring only few intermediaries to reach others. Geometric centrality, on the other hand, employs geometric mean while calculating the length of the shortest paths. This avoids domination of outliers on the result and normalises the ranges. Harmonic centrality is inspired by the harmonic mean which can be defined as the reciprocal of the arithmetic mean of the reciprocals. Harmonic mean tends toward the smallest numbers in the given set. This mitigates the domination of outliers and aggravates the impact of smaller numbers compared to the arithmetic mean.

To evaluate the presented approaches, we considered (Akkaya et al., 2013) as the baseline. Akkaya et al. (2013) is a distributed connectivity restoration approach which considers distance to the failed upstream node while selecting the leader node. We conducted extensive simulations with various metrics and show that the presented approaches not only minimises the movement cost, but also limits the number of relocated nodes.

The rest of the paper is organised as follows. Related work is summarised in Section 2. The assumptions and the problem definition are given in Section 3. Approaches are presented in Section 4. The performance of the proposed approach is evaluated in Section 5. The paper is concluded in Section 6.

2 Related work

Fault-tolerance provisioning is regarded as a form of topology management in WSNs (Younis et al., 2014a,b). The principal objective is adjusting the topology to sustain coverage while maintaining network connectivity. Fault-tolerance techniques, to tolerate permanent node failures, can be classified into two categories according to the resource provisioning time. Proactive approaches pursue a precautionary model where the resources are provisioned before failure. Solutions in this category exploit node redundancy to alleviate the consequences of node failures (Han et al., 2010; Hao et al., 2004). However, due to unpredictability of the damage location and scale, approaches of this type may not be able to ensure a solution especially for multiple collocated failures. Reactive solutions, on the other hand, provide demand-based real-time restoration. Presented approaches are reactive fault-tolerance solutions which can handle large scale simultaneous node failures.

Reactive schemes can be further classified into two broad groups based on the possibility of introducing additional nodes to the network. The first group assumes possible intervention to the deployment area and the placement of additional nodes (Senel and Younis, 2011; Senturk et al., 2014). The second type of reactive schemes, on the other hand, assumes availability of mobile nodes as part of the network and restructures network topology through relocating mobile nodes to restore connectivity (Abbasi et al., 2007; Akkaya et al., 2008; Younis et al., 2010). Both models can be further classified based on whether centralised or distributed recovery procedures are employed. Considering limited or no human intervention to the application area, presented approaches pursue the second reactive strategy and exploit mobility of the existing nodes to restore connectivity in a distributed manner.

Mobility-based connectivity restoration solutions pose two different challenges that need to be addressed. First, determining target locations for movement. Second, identifying nodes to be relocated. For instance, DARA (Abbasi et al., 2007) and PADRA (Akkaya et al., 2008) focus on the second issue. While DARA evaluates node degrees to identify the node for movement, PADRA determines connected dominating set (CDS) of the network and picks dominates for movement. There are approaches which focus on determining the target locations as well (Sir et al., 2011; Alfadhly et al., 2010; Senturk et al., 2012a). While Sir et al. (2011) considers infinitely many number of locations where the mobiles can be relocated, Alfadhly et al. (2010) and Senturk et al. (2012a) are motivated to reduce the number of locations where the nodes can move.

Recovery schemes can also be classified based on the addressed failure model. A single node failure model indicates the loss of one node at a time. On the other hand, the second failure model is characterised by simultaneous failure of multiple nodes. In distributed approaches, single node failures can be detected and tolerated by keeping the state information of the immediate neighbours within one hop. Nodes can generate a signal at regular intervals to indicate their presence and the interruption of such messages may infer node failures. Upon failure, failed node can be replaced by one of its neighbours to tolerate the node failure. However, this approach requires availability of the failed node's location before failure occurs. Moreover, this approach can only tolerate single node failures and recovery must be completed before the next failure in the same neighbourhood. However, this assumption may not be very realistic considering the exposure to surroundings and environmental conditions which makes simultaneous failure of multiple nodes very likely. (Abbasi et al., 2007; Akkaya et al., 2008; Younis et al., 2010) can only handle the loss of one node at a time and are not suitable for the considered problem. Sir et al. (2011), Alfadhly et al. (2010), Senturk et al. (2012a), Akkaya et al. (2013), Senturk et al. (2012b), Senturk and Akkaya (2012) and Senturk and Akkaya (2014), on the other hand, can tolerate simultaneous failure of collocated nodes similar to our solution.

Sir et al. (2011), Alfadhly et al. (2010) and Senturk et al. (2012a) are centralised solutions which assume the availability of the whole network state after failures. Obtaining such data from post-failure network in a centralised manner may be infeasible or even impossible and render such solutions inapplicable to the considered problem. Senturk et al. (2012b) presents a distributed solution by employing game theory. However, this approach assumes visual sensors/cameras to identify the existence of other partitions. Another distributed solution was presented in Senturk and Akkaya (2012). Though, obstacles and terrain elevation were considered in this solution and the primary goal was attaining the most energy efficient trajectories for movement in the expense of the increased movement distance.

Akkaya et al. (2013) and Senturk and Akkaya (2014) are also distributed approaches which focus on the first and second issues of the mobility-based connectivity restoration solutions respectively. Senturk and Akkaya (2014) employs betweenness and closeness centrality to evaluate the importance of the nodes and identify the node for movement. Akkaya et al. (2013), on the other hand, utilises ad-hoc on demand distance vector routing (AODV) (Perkins et al., 2003) algorithm to determine the stopping points of the trajectory to reach the BS through the shortest path. Considering the fact that the location and the scale of the damage cannot be known in advance, data collection (i.e., location) scope should be adjusted properly in distributed approaches. We follow the approach presented in Akkaya et al. (2013) to collect the upstream node locations to reach the BS.

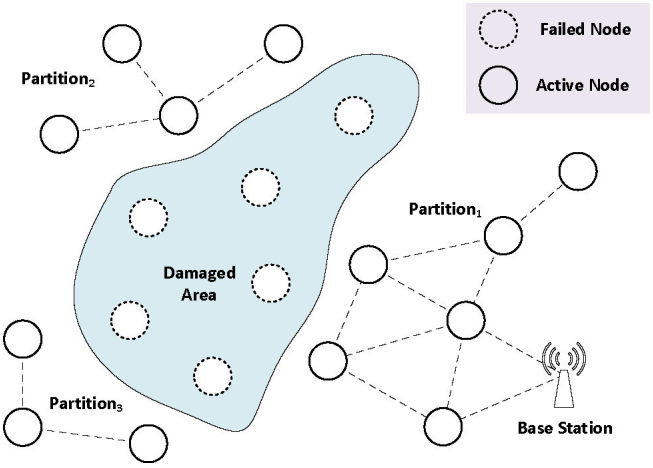
3 Preliminaries

3.1 Assumptions

We assume a connected MSN comprising a set of battery-operated mobile nodes which are randomly distributed over a region where human intervention is limited or impossible. Mobile nodes are equipped with on-board sensors to observe the area of interest and send the collected data to a stationary BS. Due to their limited transmission range, nodes form a multi-hop network to send their data to the BS in a collaborative manner. BS is assumed to act as a gateway between the network and the end user.

Unlike earlier studies which consider single node failures (Tamboli and Younis, 2010; Younis et al., 2010; Abbasi et al., 2007), we assume an arbitrary event resulting failure of multiple nodes simultaneously. Due to node failures, the network is divided into disjoint subgroups which are isolated from the rest of the network as illustrated in Figure 1. The damage is assumed to occur at a random time but after nodes establish their paths to the BS. BS is assumed to be free from failures.

Figure 1 Limited transmission range enforces forming a multi-hop mesh network to pass the data in a collaborative manner through the multi-hop network. Nodes in $Partition_2$ and $Partition_3$ are still operational but isolated from the rest of the network (see online version for colours)



We assume an unattended recovery process which involves controlled mobility to adjust network topology in a reactive manner. Therefore, the nodes are assumed to be able to detect interrupted data delivery to the BS and initiate recovery (Akkaya et al., 2013). Presented approaches require availability of the nodes' initial locations which can be obtained through one of the localisation methods (Hu and Evans, 2004). Node mobility can be enabled by attaching nodes to mobile robots (Janansefat et al., 2013) unless the nodes have inherent mobility capabilities.

3.2 Problem definition

A connected MSN is given, composed of n mobiles and one stationary BS, with a transmission range of TR. At some arbitrary moment, $1 \leq f \leq n-1$ mobiles are removed from this network creating $k > 1$ partitions. Our problem can be formally defined as follows: "Given a MSN of $n-f$ mobiles, the goal is to provide a distributed solution which will ensure that k partitions will be connected forming a single network by relocating the mobiles such that the total number of nodes to be relocated and their movement cost (i.e., $\sum_{i=1}^{n-f} d_i$) are minimised where d_i is the movement distance of $node_i$".

4 Approach

When the goal is to minimise the movement cost, mobility-based connectivity restoration solutions pose two different challenges that need to be addressed: identifying subset of nodes to be relocated and determining their movement targets. To identify nodes for movement, we introduce three different heuristics based on centrality measures. To determine the target location for movement, we have utilised the approach presented in Akkaya et al. (2013). Sections 4.1 and 4.2 detail how we address respective challenges.

4.1 The 'Who?' question – identifying the leader node for recovery

To restore connectivity, we strive to restructure the network topology through node mobility. It is possible to expedite recovery through relocating multiple nodes within the same partition simultaneously. However, it is desirable to avoid redundant movement and limit the mobility to one node at a time considering the high energy cost of mechanical motion on the batteries. To select the node for movement, various schemes can be applied. Since the loss of connection on a routing path is primarily noticed by the nodes with a failed immediate upstream node, we regard such nodes as candidates for movement and select the node to be relocated among them.

In case of multiple candidates, we follow heuristics inspired by centrality measures to assess significance of the nodes within the partition in terms of connectivity. Then we pick the candidate with the least significance anticipating the least damage on the connectivity of the partition. Note that, in graph theory, centrality measures are employed to determine how central a vertex is for the network. However, importance implied by centrality can be vague which requires elaboration on the network type and application. In this paper, we are concerned with the flow of data from many vertices to one. Thus, in our case, importance infers centrality in relaying information between vertices and the BS. Also, it should be noted that the network is partitioned. Therefore, we deal with disconnected graphs and employ partition-aware closeness, geometric, and harmonic centrality measures as detailed next.

4.1.1 Partition-aware closeness centrality (PaCC)

In a connected graph, G, closeness centrality of a vertex, $u \in G$, is defined as the reciprocal of the farness (Bavelas, 1950) as given in equation (1). Farness denotes the total length of the shortest paths between vertex u and the rest of the vertices in the graph. If a vertex is located closer to other vertices, one can expect to reach the vertex within fewer hops. Hop count is a rough measure of distance between two hosts in a network and it is useful for determining the paths in the routing algorithms (e.g., AODV). To be reachable within fewer hops infers serving on the shortest path between several other vertices and the BS. Thus, we can rank the importance of the vertices in terms of the network communication by employing closeness centrality.

In the proposed approach, we assume that the network connectivity will be damaged the most severely upon the removal of a node with a high closeness centrality. Therefore, we favour nodes with the least centrality score for movement. If two candidates have the same centrality score, we pick the one with the greater id to be the leader node. It should be noted that the worst-case time complexity of calculating *PaCC* score is $\mathcal{O}(n)$. In the worst-case, $n-f$ nodes form a single partition isolated from the BS. Each candidate collects the shortest path lengths, $\delta(u,v)$, can compute the score separately, in a distributed manner. Independent from the number of candidates, $n-f-1$ distances will be considered in equation (1) rendering the time complexity $\mathcal{O}(n)$.

$$C(u) = \frac{1}{\sum_v \delta(u,v)} \tag{1}$$

4.1.2 Partition-aware geometric centrality (PaGC)

In the second approach, we employ a centrality measure based on geometric mean. Unlike arithmetic mean, geometric mean indicates the central tendency of a set of numbers by using their products as given in equation (2). Geometric mean can be primarily used for elements with multiple features of various ranges. This avoids domination of outliers and normalises the ranges. Even though, there is only one feature (i.e., distance) to consider in our case, geometric mean can still be a viable solution to avoid the domination of outliers on the result.

In equation (2), w denotes the number of vertices, v, in the partition such that $u \neq v$.The worst-case scenario features a single isolated partition with $n-f$ nodes. Hence, $w = n-f-1$, for each candidate and the time complexity of calculating *PaGC* score is $\mathcal{O}(n)$.

$$G(u) = \left(\prod_{u \neq v} \frac{1}{\delta(u,v)} \right)^{\frac{1}{w}} \tag{2}$$

4.1.3 Partition-aware harmonic centrality (PaHC)

The third approach is a centrality measure inspired by the harmonic mean. Harmonic mean is another measure of central tendency and can be defined as the reciprocal of the arithmetic mean of the reciprocals. For the given set of numbers, harmonic mean tends toward the least numbers in the set. This mitigates the domination of outliers aggravates the impact of smaller numbers compared to the arithmetic mean. Note that, for a set of positive numbers with at least one pair of non-equal values, harmonic mean is always less than arithmetic mean

and geometric mean. Arithmetic mean, on the other hand, is always the greatest of the three (Xia et al., 1999). *PaHC* score of a given node is calculated based on the formula given in equation (3).

Similarly, the time complexity of calculating *PaHC* score is $\mathcal{O}(n)$ once the shortest path lengths are available. We will discuss the distributed algorithm to collect the shortest path lengths next. Then we will investigate the messaging cost.

$$H(u) = \sum_{u \neq v} \frac{1}{\delta(u,v)} \tag{3}$$

4.1.4 The distributed algorithm

Despite variance in calculating the centrality scores, proposed heuristics require availability of the shortest path lengths (e.g., $\delta(u,v)$). To be able to assign the shortest path length between two nodes in a given graph, one should determine the shortest path between the nodes such that the sum of the weights of its constituent edges is minimised. Since the goal is minimising the movement distance, instead of hop counts, we opt to consider distance between neighbouring nodes to denote respective edge weights. Determining the shortest path in a graph is a well studied problem where distributed algorithms also exist (Henzinger et al., 2016; Nanongkai, 2014; Chakaravarthy et al., 2017). Our problem requires solving the single-source shortest path problem, in which we have to find shortest paths from a particular node, candidate, to other nodes in the partition. Note that the employed reactive approach postpones calculating the shortest paths until partitioning and therefore the calculation is only limited within the respective partitions. This scheme simplifies the solution as elaborated next.

Both Dijkstra's algorithm and Bellman-Ford algorithm can solve the single-source shortest path problem. However, Dijkstra's algorithm is centralised and requires complete view of the network topology. Thus, we employ AODV algorithm, a variant of the Bellman-Ford for mobile networks, to identify the shortest paths. Essentially, we have modified route request (RREQ) and route reply (RREP) messages in AODV. Section 4.2 explains modifications to obtain the movement trajectory in a proactive manner. This subsection, on the other hand, clarifies how the shortest path lengths are obtained in a distributed and reactive manner. Designating candidate nodes and identifying the leader node are elaborated as well. The algorithm run by candidate $c\_id \in C$ is given in Algorithm 1. The steps followed by the rest of the nodes to compute the shortest path length for a certain candidate are given in Algorithm 2.

Since the loss of the links are primarily noticed by the nodes with a failed immediate upstream node, we consider such nodes as candidates to initiate recovery and select the leader node among them. When the network connection with the immediate upstream node is lost, nodes attempt to find an alternative path to reach the BS. Those nodes are not regarded as candidates yet considering the possibility of discovering alternative paths which will render recovery non-essential. However, if cut-vertex nodes are failed, node will infer partitioning after multiple unsuccessful attempts to find an alternative path. At this point, candidates are identified and the recovery process is initiated through Algorithm 1.

Algorithm 1 discoverNewPath(c_id)

```
1: for i = {1, 2, ..., |DS(c_id)|} do
2:     Unicast('INVALIDATE', DS(c_id)_i)
3: end for
4: Timeout(T = λ × (td + pd))  ▷ wait for invalidation
5: Broadcast(c_id, loc(c_id), 0, 'DPATH', λ)
6: Update(sp_len_c_id) = Listen('SPLEN')
7: Timeout(T'=2λ × (td + pd))  ▷ wait for RREP
8: if ∃ ('RREP') then
9:     for i = {1, 2, ..., |DS(c_id)|} do
10:        Unicast('VALIDATE', DS(c_id)_i)
11:    end for
12: else
13:    s = calculateScore(sp_len_c_id)
14:    for i = {1, 2, ..., |C| − 1} do
15:        Unicast('SCORE', c_i, s)
16:    end for
17:    s' = Listen('SCORE')
18:    if ∃ s' such that (s' < s) then
19:        Revoke candidacy
20:    else
21:        Recover(c_id)
22:    end if
23: end if
```

Algorithm 2 recvDPATH(i, ('DPATH'))

```
1: if ∃ (' VALID_PATH ') then
2:     Unicast('RREP', c_id)
3: else
4:     c_id, loc(idx), tot_len, λ' = ('DPATH')
5:     tot_len += δ(loc(i), loc(idx))
6:     if tot_len < sp_len then
7:         sp_len = tot_len
8:         Unicast('SPLEN', c_id, sp_len)
9:     end if
10:    if −−λ' > 0 then
11:        Broadcast(c_id, loc(i), tot_len, 'DPATH', λ')
12:    end if
13: end if
```

Algorithm 1, primarily, invalidates downstream nodes to avoid false route replies. INVALIDATE message propagates towards downstream nodes until reaching a leaf node. Candidate waits for a certain amount of time T before proceeding with the broadcast message of the discover new path (DPATH) request. T is defined based on transmission (td) and propagation (pd) delays. After T expires, DPATH message is broadcast. DPATH message not only looks for alternative paths but also provides a means to calculate the shortest path lengths in the partition. The content of the DPATH message is given in Table 1. The message specifies the unique node ID of the candidate, c_id, considering possibility of multiple candidates to issue the DPATH message. Upon forwarding, each node indicates its

own location, $loc(idx)$, and updates the tot_len field. Note that, for $c\_id == idx$, $tot\_len = 0$. λ denotes TTL value to limit the scope of the recovery. A detailed discussion on picking the TTL value can be found in Akkaya et al. (2013).

Table 1 DPATH (Discover New Path) message content

c_id	$loc(idx)$	tot_len	'DPATH'	λ

After broadcasting the DPATH message, candidate sets timer, T', to receive a RREP message. In the mean time, candidate listens for SPLEN messages, defined in Algorithm 2, and maintains shortest path lengths in the partition. If RREP is received within T', routing path will be complemented with an alternative path. In such a case, mobility-based recovery will not be needed and the downstream nodes will be validated. Otherwise, partitioning is inferred and mobility-based recovery is pursued. To identify the leader node, centrality score will be computed based on the employed centrality measure. In order to select the candidate with the least centrality score, candidates share their scores. The candidate assumes leadership and initiates recovery unless it receives a lower centrality score despite collecting scores from the rest of the candidates.

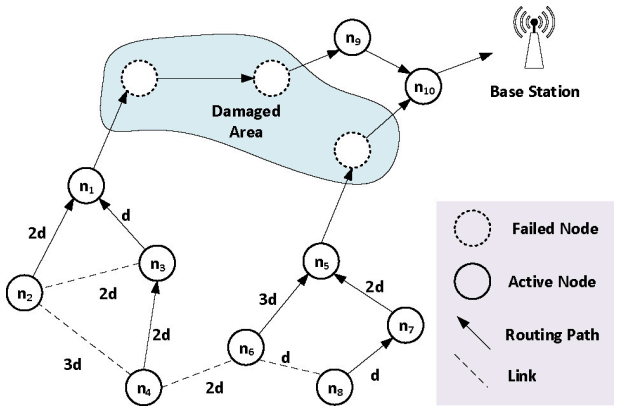
Upon receiving a DPATH message, node i applies Algorithm 2 to compute the shortest path length for candidate c_id. Recall that, node i sends a unicast message to c_id if the new path has a shorter length. sp_len, is a new column we introduced in the routing table of AODV to store the shortest path lengths of the candidates. More details on sp_len can be found in the next subsection.

4.1.5 Sample execution

A sample execution of the distributed algorithm is given in Figure 2. In the given figure, n_1 and n_5 are the first nodes to notice the failure in the damaged area since their immediate upstream nodes are failed. Before initiating mobility-based recovery, candidate nodes must ensure that no alternative paths to the BS are available and the network is partitioned. Note that, in AODV, for each destination only one route is preserved in the routing table and availability of alternative paths is possible. Before looking for alternative paths, downstream nodes must be invalidated to avoid false route replies. At this phase, n_1 invalidates n_2 and n_3, n_3 invalidates n_4, n_5 invalidates n_6 and n_7 and n_7 invalidates n_8. To discover new paths, AODV employs RREQ broadcast messages. In the proposed approach, RREQ message is complemented with the DPATH message in order to obtain shortest path lengths for the candidate nodes in a distributed manner.

We also introduce a new column, sp_len, in the routing table of AODV to store the shortest path lengths of the candidates. Upon receiving RREQ message, AODV ensures that the reverse route of the node which issued the RREQ message is in the routing table. Therefore, it is guaranteed that the candidate node which issued the DPATH message will be in the routing table of the nodes within the partition. Note that, a field to store the hop count already exists in the routing table but we prefer to exploit the actual distances. Initial value of sp_len is set to infinity.

Figure 2 Determining shortest path lengths in a distributed manner by employing modified AODV RREQ messages (see online version for colours)



n_2 and n_3 receive the DPATH message issued by n_1. They calculate their distances to candidate node n_1 as 2d and d respectively and set the sp_len column for n_1 in the routing table accordingly. Both nodes forward the message further after updating tot_len field and setting $loc(idx)$ as their own locations. n_3 and n_4 will receive the message forwarded by n_2. n_3 will drop the message since the reported tot_len (4d) will be greater than the current sp_len (d). n_4 updates tot_len as 5d and sets it as sp_len for candidate n_1. n_4 also receives DPATH message of n_1 forwarded by n_3. tot_len will be calculated as 3d which is less than the current sp_len for candidate n_1. sp_len for n_1 will be updated accordingly. The procedure proceeds in the same manner and all nodes will end up with the shortest path lengths of the candidate nodes. Each node sends a unicast message to respective candidates to report their sp_len values. Centrality scores can be calculated afterwards by employing the presented centrality metrics.

Figure 3 illustrates key nodes in a given graph when different centrality measures are employed. According to Figure 3(a), $node_8$ and $node_{16}$ have the highest centrality scores when $PaCC$ is employed and the rest of the nodes in the network have similar scores. It can be noticed from Figure 3(b) that the significance of nodes changes when $PaGC$ is employed. Lastly, Figure 3(c) demonstrates centrality scores obtained from $PaHC$.

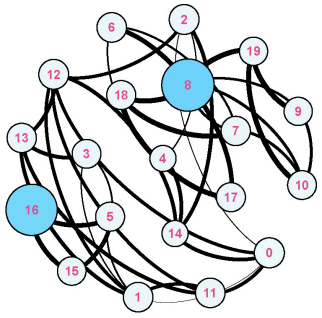
4.1.6 Algorithm analysis

Theorem 1: *Let n be the total number of remaining active nodes in the network. If m of the nodes are leaf nodes with no downstream nodes, and there are α candidates (i.e., nodes with a failed immediate upstream node) then the worst-case message complexity of the proposed approach is $\mathcal{O}(n\alpha)$.*

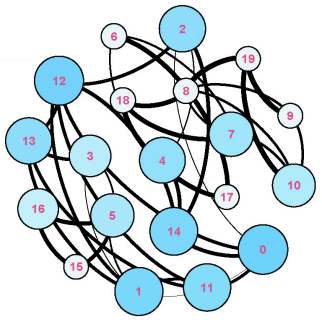
Proof: Route invalidation is initiated by the candidate node and forwarded by the downstream nodes until reaching a leaf node. For the worst-case scenario, we assume a network of two partitions where the BS comprises the first partition while the

second partition is composed of n nodes. Then, $(n - \alpha - m)$ messages will be sent. In the worst-case, $\alpha = 1$ and $m = 0$.

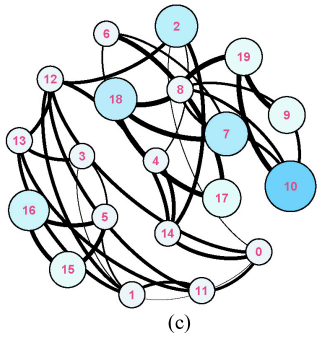
Figure 3 Identified key nodes vary based on the employed centrality metric. Increased edge thickness denotes higher distance between corresponding nodes. Node size and darkness indicate higher centrality score: (a) key nodes in the network when PaCC is employed; (b) key nodes in the network when PaGC is employed and (c) key nodes in the network when PaHC is employed (see online version for colours)



(a)



(b)



(c)

Partition detection and the shortest path length calculation are handled through DPATH messages. Again, candidate nodes initiate DPATH messages which are broadcast up to λ times until reaching a leaf node or a node with a valid path. In the worst-case, failure of a single node isolates the remaining n active nodes from the BS. For α candidates, $n - 1$ nodes will be involved in broadcasting DPATH messages for nodes in the form of a tree with depth λ in the worst-case. Therefore, the message cost for DPATH will be $(n - 1) \times \alpha \times \lambda$ in the worst-case. Since the network is partitioned, there will not be any RREP messages in this case.

SPLEN is a unicast message destined to the respective candidate node. A node may need to send multiple SPLEN messages to the same candidate, however, this is limited with the condition of discovering a shorter path. In the worst case, message complexity for SPLEN will be $\alpha \times (n - 1)$.

SCORE is a unicast message sent from candidates to other candidates. In the worst-case, there will be $\alpha \times (\alpha - 1)$ messages considering α candidates in the network.

During cascaded movement, leader broadcasts MOVING and MOVED messages (Akkaya et al., 2013) respectively before and after each movement. MOVING message notifies the neighbours regarding relocation of the leader. MOVED message is employed to discover active nodes around the new location after movement. Both messages are broadcast with TTL=1 and in the worst-case, all nodes move and a total of $2n$ messages are sent.

In the worst-case, for recovery, a total of $(n - 1) + (n - 1) \times \alpha \times \lambda + \alpha \times (n - 1) + \alpha \times (\alpha - 1) + 2n$ messages are sent. This is $\mathcal{O}(n\alpha\lambda)$ and converges to $\mathcal{O}(n\alpha)$ for small values of λ. This message complexity is same with the baseline. □

4.2 The 'Where?' question – determining the location for the movement target

When the application area is modelled as two dimensional Euclidean plane with coordinates defined by the real coordinate space, there will be infinitely many number of locations to be considered as the movement target. Obtaining the optimal solution is regarded as NP-Hard (Sir et al., 2011) and we will pursue a heuristic to limit the possible target locations instead. Note that, single node failures can be tolerated by replacing the failed node with one of the neighbours. This idea can be adapted to simultaneous node failures as well. Leader node initiates recovery by replacing its failed immediate upstream node and looks for a functional node with a valid route to the BS. Since the damage is now scaled to several nodes, the next functional node may be multiple hops away. In such a case, movement of the leader will not be sufficient to restore connection. To ensure recovery, movement should proceed with the next step.

To determine the movement trajectory, we follow the approach presented in Akkaya et al. (2013). The idea suggests determining the shortest path to the BS for each node and then using this path as the movement trajectory upon needed. This approach requires availability of the locations for the corresponding nodes in the path. Since the network is initially connected, such a path along with the locations can be easily obtained with a slight modification in the routing algorithm. If the localisation is enabled, we can use the route reply messages (as in AODV) to efficiently obtain location information of the upstream nodes in the shortest path. When the route reply messages are forwarded towards the destination, each node adds its own ID and location in the message. If a route reply message triggers an update in the routing table of the intermediate nodes which forward the message or the destination node, which initiates the request, upstream path in the route reply message is copied by the node so that the full upstream path is obtained.

If the route construction phase is over before the partitioning, this approach ensures recovery. In the worst-case, node moves until reaching the BS. If the movement of the

leader node is not sufficient to restore the connection, recovery procedure is sustained with the new leader node in an iterative manner. One node is replaced at a time until two partitions are federated.

5 Experimental evaluation

In this section, we explain the experiment setup and define performance metrics initially. Then we identify the baseline. For each performance metric, the results are demonstrated and discussed.

5.1 Experiment setup

The performance of the presented approaches have been evaluated in a simulated environment. An application area of 600×600 m is assumed where the nodes are deployed in a random fashion. Nodes and the BS have a transmission range of 30 m and form a connected network initially. To observe the correlation of the node density and the damage scale on the recovery cost, two sets of topologies are generated. In the first set, the number of partitions is set to 3 and the number of nodes is varied from 40 to 120. In the second set, the number of nodes is set to 60 and the number of partitions is varied from 2 to 5. For each case, 50 different topologies are created and tested for significance and the average is reported.

The partitions are created as follows: When the topologies are generated, we formed connected components as many as the desired partition count and placed them apart from each other with a distance of more than the transmission range. The nodes were shared evenly among the partitions. Once the partitioned network is deployed, we connected the partitions by applying the approach presented in Senel and Younis (2011) and obtained a single connected MSN. Senel and Younis (2011) is a relay placement solution with a goal of connecting partitions while minimising the number of relays to be used. The output of the mentioned approach is a set of locations where the relay nodes can be positioned so that the network will be connected. We deploy nodes to the provided locations and mark them to be failed. In the experiments, when the nodes determine their paths to the BS, we assume a random catastrophic event and remove the marked nodes from the network to form the partitions. The average number of nodes to be removed from the network can be found in Tables 2 and 3 for varying network size and damage scale respectively.

Table 2 The average number of failed nodes to form 3 partitions when the node density is varied

No. of nodes	*No. of failed nodes*
40	9.76
60	8.82
80	7.42
100	6.42
120	6.22

Table 3 The average number of failed nodes to form varying number of partitions when the network size is fixed to 60 nodes

No. of partitions	*No. of failed nodes*
2	4.46
3	8.82
4	11.24
5	13.64

5.2 Performance metrics

- *Total movement distance*: This metric is to measure the total distance travelled by the nodes involved in recovery. Movement incurs excessive energy consumption and shortens the node's lifetime. Thus, movement cost should be minimised in order to extend the overall network lifetime.
- *Participation to recovery*: This metric indicates the number of mobiles involved in recovery. Depending on the damage scale, movement of a single node may not be sufficient and cascaded movement may be required. It is preferred to limit the number of nodes involved in the cascaded movement process to enhance the network lifetime.
- *Maximum cascaded movement distance*: This metric points the largest cumulative movement distance to recover a partition. Based on the velocity of the mobile, this metric implies maximum delay occurred due to cascaded movement. Thus, it is desirable to limit the maximum cascaded movement distance.
- *Coverage*: This metric denotes total area sensed by the nodes in network. The higher the coverage the better the data accuracy and fidelity from the region.

5.3 Baseline

We employ the leader node selection scheme of Akkaya et al. (2013) as the baseline. Throughout the rest of the paper, *DiBA* (Distance-based Approach) is used to represent the baseline. When multiple candidates are available, *DiBA* considers the distance between the candidate and its failed immediate upstream node. The candidate with the shortest distance is selected as the leader node to initiate recovery. Note that the leader may have to continue movement in an iterative manner until discovering a node with sustained connection. Unless the movement of the leader is sufficient, *DiBA* applies cascaded movement and pursue recovery with the next leader in the partition.

5.4 Performance results

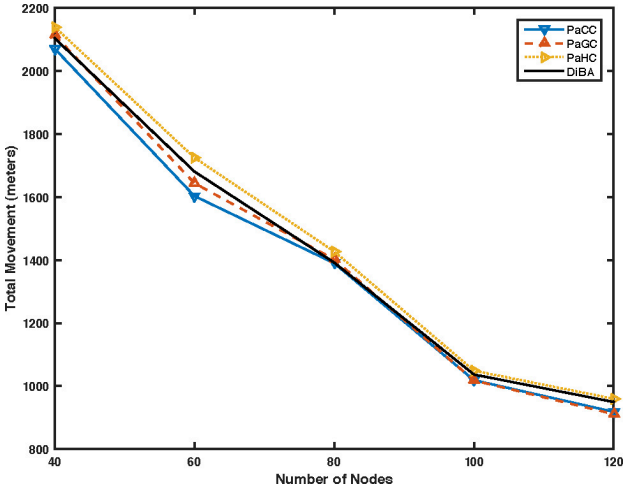
5.4.1 Total movement distance

Connectivity restoration costs in terms of total movement distance are presented in Figures 4 and 5, respectively, for various network size and damage scale. Figure 4 indicates the

decrease in distance to be travelled to ensure recovery when the number of nodes in the network is increased. This negative correlation can be attributed to the increased node density (i.e., redundancy) considering the fixed size of the application area. Consequently, recovery cost declines with the improved availability of alternative paths for recovery. Distance between the partitions is also expected to be shorter due to the increased node density for the given application area.

Figure 4 suggests that the minimum recovery cost can be attained by employing *PaCC* in sparse networks and *PaGC* in dense networks. *PaCC* and *PaGC* not only scales well but also outperforms the baseline. *PaHC*, on the other hand, is outperformed by the baseline especially in sparse networks.

Figure 4 Total movement distance with varying number of nodes (see online version for colours)



It can be observed from Figure 5 that the total movement distance increases when the damage scale is extended. This is expected due to the increased demand for recovery from additional partitions. *PaCC* ensures recovery with the least cost for all damage levels. *PaGC* outperforms the baseline when the number of partitions is less than 4. They perform similar for increased partition counts. *PaHC* is outperformed by the baseline again.

5.4.2 *Participation to recovery*

Movement of the leader node initiates recovery. However, connectivity may not be restored by a single node movement. Furthermore, movement of the leader node may cause further partitions in the network. In such a case, recovery should be proceeded with further attempts involving more nodes in a cascaded manner. Considering the cost of mobility, it is desirable to limit the number of nodes involved in recovery. In this subsection, we report the number of mobiles participated to recovery. Figures 6 and 7 depict the results for varying node densities and damage levels, respectively.

Figure 6 reveals that the scope of the recovery decreases with the increased node density. This can be attributed to the increased likelihood of discovering alternative nodes for connection when the network size is larger. Consequently, the number of nodes involved in recovery declines. *PaCC*, *PaGC*, and *PaHC* outperform the baseline in all node densities. The baseline requires the most nodes involved in recovery.

Figure 5 Total movement distance with varying number of partitions (see online version for colours)

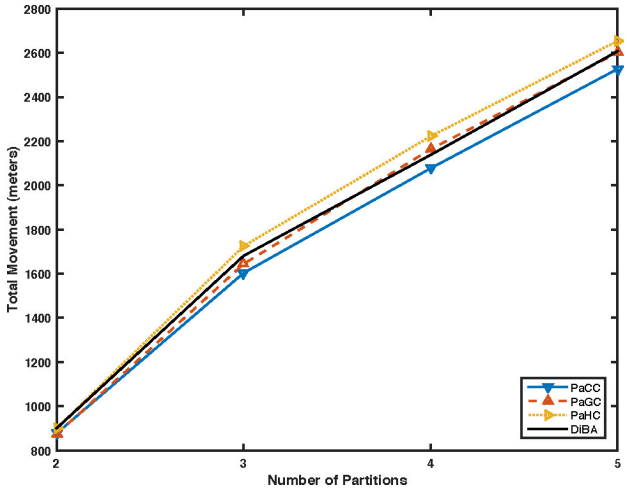
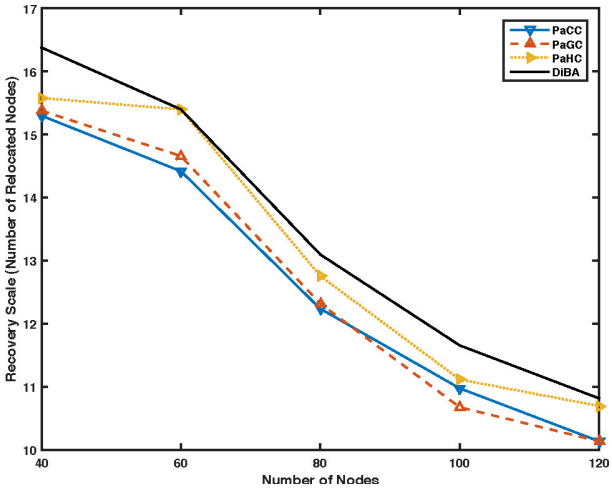


Figure 6 The number of relocated nodes with varying number of nodes (see online version for colours)



It can be observed from Figure 7 that the scope of the recovery increases when the damage scale is extended. This is expected due to the increased number of partitions which needs to be recovered. *PaCC*, *PaGC*, and *PaHC*, again, outperform the baseline and require the least nodes to be involved in recovery.

5.4.3 *Maximum cascaded movement*

For the topologies with two partitions, this metric denotes total movement distance. When there are more than two partitions, longest cascaded movement indicates the recovery delay considering concurrent recovery effort in all partitions. Therefore, we evaluate the approaches in terms of the maximum cascaded movement distance to obtain recovery delay. The results are depicted in Figures 8 and 9 for varying node density and damage scale, respectively.

Figure 8 indicates that the maximum cascaded movement distance declines for all approaches when the node density is

increased. This is expected due to the reasons justified earlier. *PaCC* and *PaCC* outperform the baseline in all node densities. *PaHC* is outperformed by the baseline in sparse networks, but performs similar when the number of nodes is 80 or more.

Figure 7 The number of relocated nodes with varying number of partitions (see online version for colours)

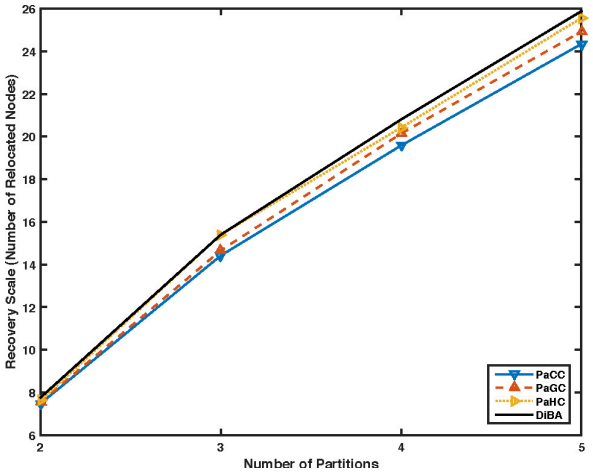
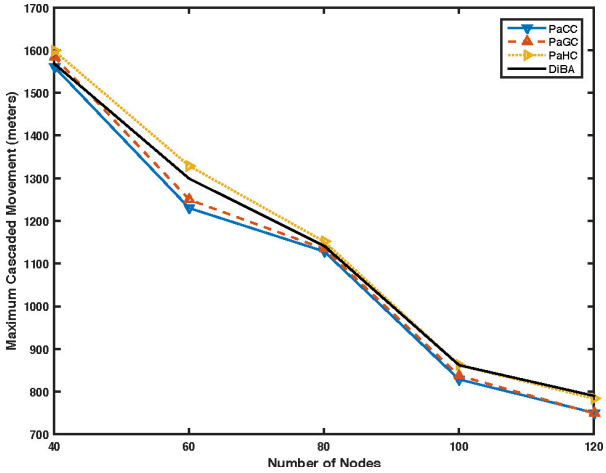


Figure 8 Maximum cascaded movement distance with varying number of nodes (see online version for colours)



As expected, Figure 9 denotes that the maximum cascaded movement distance increases for all approaches when the damage scale is extended. *PaCC* provides the least cost in all damage levels. *PaGC* also outperforms the baseline up to 5 partitions but performs similar to *PaHC* afterwards. Despite the almost constant cost increase up to 4 partitions, recovery cost reaches a threshold and the cost increase rate declines next. This is analogous to the average distance between the partitions. One can claim that the maximum cascaded movement distance can even decline if the number of partitions is increased further while the size of the application area is fixed.

5.4.4 Coverage

In order to determine how each approach improves the coverage after the partitioning, we conducted experiments with varying number of nodes and partitions as illustrated in Tables 4 and 5 respectively. According to Table 4, coverage increases with the increased number of nodes in the network. However, the increase in coverage is not proportional to the increase in node density. This can be attributed to the redundancy in coverage. This is the case when a region is covered by multiple nodes. On the other hand, coverage declines dramatically after node failures. Note that nodes, whether functional or not, do not contribute to the network coverage if they do not reside within the same partition with the BS. To remedy the problem, we apply presented approaches. Table 4 denotes that *DiBA* outperforms the presented approaches. On the other hand, *PaCC* provides better coverage than *PaGC* and *PaHC*. For 40 nodes, *DiBA* recovers up to 86% of the initial coverage while *PaCC* provides a recovery of 84%. When the number of nodes is increased to 120 nodes, *DiBA* and *PaCC* provide 97% and 96% coverage recovery respectively.

Figure 9 Maximum cascaded movement distance with varying number of partitions (see online version for colours)

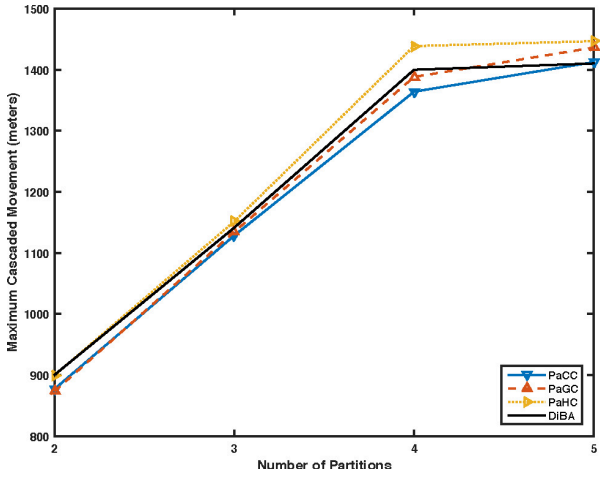


Table 5 indicates a slight increase in the coverage with the increased partition count. Considering the fixed network density, network coverage is expected to be almost constant. But in the initial network topology, we consider the coverage of the nodes to be failed as well and the number of such nodes increases with the increased partition count. As expected, post-failure network coverage declines with the increased partition count. *DiBA*, again, outperforms the presented approaches and *PaCC* provides better coverage than *PaGC* and *PaHC*. For 2 partitions, *DiBA* and *PaCC* provide 96% and 95% coverage recovery respectively. When the number of partitions is increased to 5, coverage recovery declines to 87% and 85% for *DiBA* and *PaCC* respectively.

Table 4 Network coverage (m^2) with 3 partitions and varying number of nodes when different approaches are applied

No. of nodes	*40*	*60*	*80*	*100*	*120*
Initial	28,377	36,532	45,174	53,814	62,002
Post-failure	7129	10,430	13,464	16,686	19,743
PaCC	23,809	32,850	42,195	51,267	59,659
PaGC	23,673	32,743	42,079	51,180	59,574
PaHC	23,613	32,688	41,908	50,970	59,431
DiBA	24,287	33,838	42,981	51,918	60,280

Table 5 Network coverage (m^2) with 60 nodes and varying number of partitions when different approaches are applied

No. of partitions	*2*	*3*	*4*	*5*
Initial	33,763	36,532	39,535	42,103
Post-failure	15,705	10,430	8,024	6555
PaCC	31,911	32,850	34,632	35,863
PaGC	31,863	32,743	34,408	35,715
PaHC	31,774	32,688	34,303	35,603
DiBA	32,356	33,838	35,458	36,541

6 Conclusions

In this paper, we considered the problem of connectivity restoration in partitioned MSNs. To ensure recovery, we have exploited controlled node mobility. In order to minimise the movement cost, mobility-based connectivity restoration solutions require addressing two different challenges. First, identifying the subset of nodes to be relocated. Second, determining target locations for movement. To address the first problem, we have presented three different heuristics based on centrality measures. The idea is identifying key nodes within the network as in social network analysis and selecting the least important node for relocation. The first approach, *PaCC*, employs closeness centrality which considers the total length of the shortest paths to the rest of the nodes in the partition. The other two approaches, *PaGC* and *PaHC*, are inspired by the geometric and harmonic means respectively. As the experiments reveal, *PaCC* and *PaGC* ensure recovery with the least costs and outperform the baseline, *DiBA*, in terms of total movement distance, maximum cascaded movement distance, and the number of nodes involved in recovery.

Acknowledgement

This work was supported by the Scientific and Technical Research Council of Turkey (TUBITAK) under Grant No. EEEAG-117E050.

References

Abbasi, A.A., Akkaya, K. and Younis, M. (2007) 'A distributed connectivity restoration algorithm in wireless sensor and actor networks', *Proceedings of the 32Nd IEEE Conference on Local Computer Networks, LCN '07*, IEEE Computer Society, Washington DC, USA, pp.496–503.

Akkaya, K., Senturk, I.F. and Vemulapalli, S. (2013) 'Handling large-scale node failures in mobile sensor/robot networks', *Journal of Network and Computer Applications*, Vol. 36, No. 1, pp.195–210.

Akkaya, K., Thimmapuram, A., Senel, F. and Uludag, S. (2008) 'Distributed recovery of actor failures in wireless sensor and actor networks', *2008 IEEE Wireless Communications and Networking Conference*, pp.2480–2485.

Alfadhly, A., Baroudi, U. and Younis, M. (2010) 'Optimal node repositioning for tolerating node failure in wireless sensor actor network', *2010 25th Biennial Symposium on Communications (QBSC)*, pp.67–71.

Bavelas, A. (1950) 'Communication patterns in taskâŁŁoriented groups', *The Journal of the Acoustical Society of America*, Vol. 22, No. 6, pp.725–730.

Chakaravarthy, V.T., Checconi, F., Murali, P., Petrini, F. and Sabharwal, Y. (2017) 'Scalable single source shortest path algorithms for massively parallel systems', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 28, No. 7, pp.2031–2045.

Han, X., Cao, X., Lloyd, E.L. and Shen, C.-C. (2010) 'Fault-tolerant relay node placement in heterogeneous wireless sensor networks', *IEEE Transactions on Mobile Computing*, Vol. 9, No. 5, pp.643–656.

Hao, B., Tang, H. and Xue, G. (2004) 'Fault-tolerant relay node placement in wireless sensor networks: formulation and approximation', *2004 Workshop on High Performance Switching and Routing, 2004. HPSR*, pp.246–250.

Henzinger, M., Krinninger, S. and Nanongkai, D. (2016) 'A deterministic almost-tight distributed algorithm for approximating single-source shortest paths', *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing', STOC '16*, ACM, New York, NY, USA, pp.489–498.

Hu, L. and Evans, D. (2004) 'Localization for mobile sensor networks', *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, MobiCom '04*, ACM, New York, NY, USA, pp.45–57.

Janansefat, S., Akkaya, K., Senturk, I.F. and Gloff, M. (2013) 'Rethinking connectivity restoration in wsns using feedback from a low-cost mobile sensor network testbed', *2013 IEEE 38th Conference on Local Computer Networks Workshops (LCN Workshops)*, pp.108–115.

Nanongkai, D. (2014) 'Distributed approximation algorithms for weighted shortest paths', *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing, STOC '14*, ACM, New York, NY, USA, pp.565–573.

Natalizio, E. and Loscrí, V. (2013) 'Controlled mobility in mobile sensor networks: advantages, issues and challenges', *Telecommunication Systems*, Vol. 52, No. 4, pp.2411–2418.

Perkins, C., Belding-Royer, E. and Das, S. (2003) *Ad Hoc On-Demand Distance Vector (AODV) Routing*, RFC 3561, RFC Editor.

Senel, F. and Younis, M. (2011) 'Optimized connectivity restoration in a partitioned wireless sensor network', *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pp.1–5.

Senturk, I. and Akkaya, K. (2012) 'Energy and terrain aware connectivity restoration in disjoint mobile sensor networks', *2012 IEEE 37th Conference on Local Computer Networks Workshops (LCN Workshops)*, pp.767–774.

Senturk, I.F. (2017) 'A prescient recovery approach for disjoint MSNs', *2017 IEEE International Conference on Communications (ICC)*, pp.1–6.

Senturk, I.F. and Akkaya, K. (2014) 'Connectivity restoration in disjoint wireless sensor networks using centrality measures', *2014 IEEE 39th Conference on Local Computer Networks Workshops (LCN Workshops)*, pp.616–622.

Senturk, I.F., Akkaya, K. and Yilmaz, S. (2014) 'Relay placement for restoring connectivity in partitioned wireless sensor networks under limited information', *Ad Hoc Networks*, Vol. 13, Part B, pp.487–503.

Senturk, I.F., Akkaya, K. and Senel, F. (2012a) 'An effective and scalable connectivity restoration heuristic for mobile sensor/actor networks', *Global Communications Conference (GLOBECOM), 2012 IEEE*, pp.518–523.

Senturk, I., Yilmaz, S. and Akkaya, K. (2012b) 'Connectivity restoration in delay-tolerant sensor networks using game theory', *Int. J. Ad Hoc Ubiquitous Comput.*, Vol. 11, Nos. 2–3, pp.109–124.

Sir, M.Y., Senturk, I.F., Sisikoglu, E. and Akkaya, K. (2011) 'An optimization-based approach for connecting partitioned mobile sensor/actuator networks', *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp.525–530.

Tamboli, N. and Younis, M. (2010) 'Coverage-aware connectivity restoration in mobile sensor networks', *Journal of Network and Computer Applications*, Vol 33, No. 4, pp.363–374.

Vecchio, M., Viana, A.C., Ziviani, A. and Friedman, R. (2010) 'Deep: density-based proactive data dissemination protocol for wireless sensor networks with uncontrolled sink mobility', *Computer Communications*, Vol. 33, No. 8, pp.929–939, Special Section on Hot Topics in Mesh Networking.

Xia, D-F., Xu, S-L. and Qi, F. (1999) 'A proof of the arithmetic mean-geometric mean-harmonic mean inequalities', *RGMIA Research Report Collection*.

Younis, M., Lee, S. and Abbasi, A.A. (2010) 'A localized algorithm for restoring internode connectivity in networks of moveable sensors', *IEEE Transactions on Computers*, Vol. 59, No. 12, pp.1669–1682.

Younis, M., Lee, S., Senturk, I.F. and Akkaya, K. (2014a) *Topology Management Techniques for Tolerating Node Failure*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp.273–311.

Younis, M., Senturk, I.F., Akkaya, K., Lee, S. and Senel, F. (2014b) 'Survey topology management techniques for tolerating node failures in wireless sensor networks: a survey', *Comput. Netw.*, Vol. 58, pp.254–283.