

Mesh-Based Sensor Relocation for Coverage Maintenance in Mobile Sensor Networks

Xu Li¹, Nicola Santoro¹, and Ivan Stojmenovic²

¹ SCS, Carleton University, Ottawa, Canada
{xlii,santoro}@scs.carleton.ca

² EECE, Engineering, University of Birmingham, UK
ivan@site.uottawa.ca

Abstract. Sensor relocation protocols can be employed as fault tolerance approach to offset the coverage loss caused by node failures. We introduce a novel localized structure, *information mesh*, for publishing and retrieving node location information. Based on the structure, we propose a Mesh-based Sensor Relocation Protocol (MSRP) for mobile sensor networks. MSRP maintains a sensor network's sensing coverage by replacing failed sensors with nearby redundant ones using near optimal time delay and balanced energy consumption. Simulation indicates that it guarantees closest node replacement with high probability (> 96%) and with considerably low message overhead. We show that MSRP is superior to the existing sensor relocation schemes for its localized message transmissions and optimal (constant) per node storage load.

1 Introduction

Sensing coverage (or coverage) is an important QoS factor. It is measured by the overall area that a sensor network is currently monitoring. The larger the coverage, the better service the network can provide. In order to maintain quality of service, a sensor network must be able to preserve its coverage in the presence of node failures. In static sensor networks, distributing a large number of redundant sensors is the only way to tolerate node failures. Relevant research [3] concentrates mainly on how to schedule sensor activities to save energy without jeopardizing coverage. However, in mobile sensor networks, node mobility can be exploited to facilitate coverage maintenance. That is, it can be used to deploy a moderate number of redundant sensors and strategically relocate them to fill the position of failed nodes. This type of movement-assisted coverage maintenance approaches are called *sensor relocation* and constitute the focus of this article.

In our work, we consider a connected mobile sensor network deployed in plane, where uncoverable obstacles such as hills and lakes may exist. We assume that the network has achieved a full coverage over the coverable area in the sensor field, through a sensor self-deployment algorithm [4,6], after its initial placement. The nodes that constitute the coverage always remain active and are called *active nodes* (or *A-nodes*). We also assume that some predefined *redundant nodes* (or *R-nodes*) are scattered in the network at random. R-nodes run a sleep/wakeup

protocol to save energy and do not contribute to network connectivity. Both A-nodes and R-nodes stay static unless they are requested to move. All the nodes are homogeneous. They are aware of their own location (an (x,y) coordinate) by a localization system and know about their neighbors' position through lower level protocols. Their communication radius cR is at least twice as large as their sensing radius sR . Any node may fail at any time.

Our research goal is to develop, based on the above network model, a sensor relocation protocol that can maintain a network's coverage by relocating nearby R-nodes to the position of failed A-nodes. Note that, in a disconnected network, there is no guarantee that failed A-nodes are successfully replaced. For this reason, we additionally assume the network always remains connected in spite of node failures. Further, message collision and transmission errors can also affect the effectiveness of solution protocols. However, since these are MAC layer issues and beyond the scope of this paper, we assume perfect wireless communication channels so that we can concentrate on the sensor relocation problem itself.

In this paper, we propose a mesh-based sensor relocation protocol (MSRP) to solve the sensor relocation problem. With zero pre-knowledge of the sensor field, MSRP accomplishes the following two tasks: finding a nearby redundant sensor for sensor replacement (*replacement discovery task*) and moving the discovered redundant sensor to replace a failed one (*node relocation task*).

MSRP accomplishes the replacement discovery task by a Distance-Sensitive Node Discovery algorithm (DSND). Based on some dynamically determined R-node proxies, DSND constructs a localized *information mesh* with constant per node storage load. Upon an A-node failure, the A-node neighbors of the failed node find a nearby R-node proxy via the information mesh and take the proxy node's nearest delegated R-node as the failed node's replacement. The message complexity of DSND is never larger, in fact much smaller in average case, than that of the node discovery methods of the existing relocation protocols.

MSRP fulfills the node relocation task by a shifted node relocation method. That is, it builds a path between a failed A-node and a discovered R-node and then shifts all the nodes' position along the path toward the failed A-node. This method uses a novel localized relocation path discovery mechanism and generates constant relocation delay and balanced energy consumption. Two shifted relocation methods have been presented in [12, 5]. But, they both have weakness in their relocation path discovery part, when compared with MSRP.

In the rest of the paper, we first summarize some related work in Sec. 2. Then, in 3, we present algorithm DSND and evaluate its performance through simulation. Afterward, we propose protocol MSRP in Section 4 and discuss its implementation issues in Sec. 5. Space limitation does not allow to present all details and proofs. Readers can find them in the full version of this article in [7].

2 Related Work

Wang, Cao and Porta [11] presented a proxy-based sensor relocation protocol (referred to as WCP) for the sensor networks composed of both static nodes and

Table 1. Protocol comparison

	MSRP	WCP [11]	WCPZ [12]	ZONER [5]
Protocol Nature	localized	quasi-distributed	quasi-distributed	localized
ZERO Pre-knowledge	yes	yes	no	yes
Guaranteed Node Replacing	yes	no	no	yes
Constant Relocation Latency	yes	no	yes	yes
Constant Per-node Storage Load	yes	no	no	no
Replacement Discovery Method	mesh	flooding	quorum	quorum
Closest Replacement Discovery	no	no	no	yes
Message Complexity	$O(\psi(G))$	$O(n'n)$	$O(n'\sqrt{n})$	$O(n'\sqrt{n})$
Node Relocation Method	shifted	direct	shifted	shifted
Guaranteed Relocation Path Discovery	yes	N/A	no	yes
Energy-aware Relocation Path	yes	N/A	yes	no

n and n' denote the number of A-nodes and the number of R-nodes, respectively; $\nu + \sqrt{n} \leq \psi(G) \leq \text{Min}\{\nu\sqrt{n}, n\}$ where $\nu \leq \text{Min}\{n', n\}$; $O(\psi(G))$ can be much less than $O(n)$ in terms of order of magnitude when $\nu > \sqrt{n}$.

mobiles. Static nodes construct a Voronoi diagram and bids closest mobiles to fill the sensing holes in their Voronoi polygons. Mobile nodes move to large holes from small ones. WCP relies on flooding for replacement discovery and uses a direct relocation method that can generate inconstant relocation delay.

Wang, Cao, Porta and Zhang [12] presented a grid-quorum-based relocation protocol (referred to as WCPZ) for mobile sensor networks. The network field is partitioned into a 2-D grid. In each grid cell, a node as cell head run the quorum-based location service [9, 8] over the grid to find replacement for local failed sensors. Then the discovered replacement is relocated along a carefully selected path in a cascaded (shifted) way. WCPZ requires pre-knowledge of the sensor field and may fail in the presence of void grid cells.

Li and Santoro [5] presented a zone-based relocation protocol (ZONER) for mobile sensor networks with pre-deployed redundant sensors. Similar to WCPZ, it is also a variant of the quorum-based location service, and it employs the shifted relocation method as well. However, it requires no pre-knowledge of the sensor field and guarantees replacement discovery (by resorting to face routing) and node replacing, and therefore it is superior to WCPZ [12].

To our knowledge, the above three sensor relocation protocols are the only existing ones for coverage maintenance in the literature. Beside the weaknesses stated above, they share a common drawback, i.e., inconstant per node storage load. The properties of these protocols are listed, in comparison with our new protocol MSRP, in Tab. 1. Their detailed analysis can be found in [7].

3 Distance-Sensitive Node Discovery

Each R-node spontaneously takes the nearest neighboring A-node as *proxy*. In case of tie, nodal relative position can be used to help make decisions. A R-node has one and only one proxy, while a proxy may be shared by multiple R-nodes.

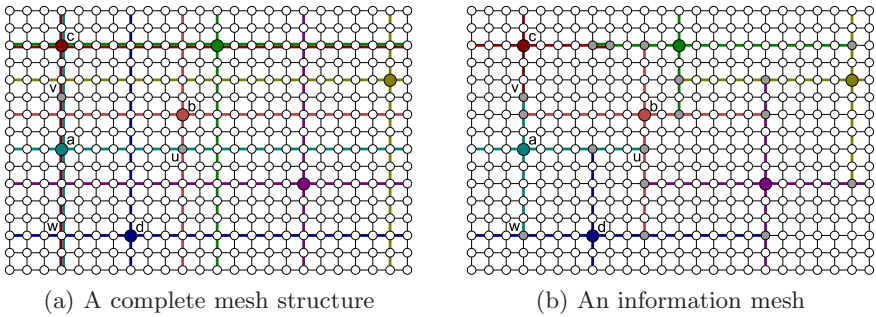


Fig. 1. Information mesh construction in a grid sensor network

Proxy nodes record the location of their delegated R-nodes and together construct an information mesh over the network. When a nearby R-node is wanted, an A-node just need to find a proxy node in its vicinity. The core of DSND consists of two parts: *information mesh construction* and *proxy node lookup*. For easy understanding, in the following, we present the two key components first in well-structured grid networks and then in arbitrary network scenarios, ignoring practical impact factors and implementation details.

3.1 Grid Sensor Networks

In a grid sensor network, A-nodes are placed exactly at the intersection points of a grid structure. Any A-node can find out its own role in the grid structure just by counting the number of its A-node neighbors. Consider only the residing rows and columns of the proxy nodes in the network. They intersect each other and form a mesh structure, as illustrated in Fig. 1(a). In this figure, R-nodes are not displayed; proxy nodes are represented by big colorful dots, and their residing rows and columns are highlighted by the corresponding color. If each proxy node distributes its own location information among the A-nodes along its residing row and column, then this mesh structure distributedly stores the location information of all the proxy nodes.

Let us examine the mesh structure shown in Fig. 1(a). Proxy node *c* is closer to the area above the mid-point A- *v* between itself and the vertically collinear proxy node *a*, and thus it (essentially, its delegated R-nodes) has relatively high priority to be discovered by the A-nodes in that area. In addition, proxy node *b* might be a better choice for the A-nodes located in its right-side area than proxy node *a*. In these cases, *a* does not need to distribute its location information in those areas. Similar argument can be made against other proxy nodes. According to this observation, we define the following **blocking rule**:

For an A-node u shared by the residing rows/columns of two different proxy nodes a and b , it stops the further propagation of a 's location information, if and only if $(|ua| > |ub|) \vee (|ua| = |ub| \wedge \neg Colline(a, b) \wedge Horizontal(a, b)) \vee (|ua| = |ub| \wedge Colline(a, b))$, where $Colline(a, b)$ and $Horizontal(a, b)$ denote the case that a and b are (vertically or horizontally) collinear and the case that the

involvement of b is along the horizontal direction, respectively. And, when this blocking happens, we say “ b blocks a at u ”.

The application of above blocking rule can lead to the merger of adjacent mesh cells and result in a pruned mesh structure, i.e., *information mesh*, Figure 1(b), where gray dots represent the A-nodes at which the blocking rule applies, shows the information mesh corresponding to the complete mesh structure in Fig. 1(a). Due to asynchronous execution and proxy-role change, information inconsistency can occur in the information mesh. DSND uses *revocation processes* to eliminate inconsistent information. Please refer to [7] for details.

Consider an arbitrary A-node a . Define the *Home Cell* of a as the mesh cell where a is located in or the aggregation of the mesh cells which it is adjacent by. And, let the *Set of Proxies in Vicinity (SPV)* of a be the set of proxy nodes whose residing grid rows/columns form the home cell of a . Then for a , its objective of proxy lookup is to identify the location of the nearest proxy node, referred to as *target proxy*, in its SPV. Note that there exist examples where a node’s target proxy is not a globally closest proxy.

Suppose that a is residing in a cell of the information mesh. When it wants to find its target proxy, it just inquires the A-nodes along its residing row and column in the grid structure in four directions. By this means, a is able to reach all the mesh edges constituting its home cell and get the location of the proxy nodes recorded on those edges. After that, it can find its target proxy simply through a local comparison. If there exist no R-node (i.e., no proxy) in the network, a will be aware after it reaches the border of the network in each of its query directions. Because the query paths form a cross, this type of proxy lookup method is called *cross lookup*. Cross lookup can also be applied when a is located on the information mesh. In this case, a inquires along its residing mesh edges and stops at their farthest ends on its home perimeter.

Let us denote the underlying grid sensor network by G . And, let $\nu(G)$ (or simply ν) represent the number of proxy nodes in G . By definition, $\nu(G) \leq \text{Min}\{n, n'\}$ where n and n' are respectively the number of A-nodes and the number of R-nodes. In addition, denote the information mesh constructed on top of G by $\mathcal{IM}(G)$ (or simply by \mathcal{IM}), and define the *extension* $\eta(\mathcal{IM})$ (or η for short) of \mathcal{IM} as the length summation of all the edges in \mathcal{IM} . Then we have the following lemmas and theorems, whose proofs can be found in [7].

Lemma 1. *In a square G , $\eta \in O(\text{Min}\{\nu\sqrt{n}, n\})$.*

Lemma 2. *In a square G , $\eta = \Omega(\nu + \sqrt{n})$.*

Theorem 1. *In a square G , the message complexity of information mesh construction is $O(\psi(G))$ where $\nu + \sqrt{n} \leq \psi(G) \leq \text{Min}\{\nu\sqrt{n}, n\}$.*

Theorem 2. *In a square G , the message complexity of cross lookup is $O(\sqrt{n})$.*

Theorem 3. *In an arbitrary G , \mathcal{IM} generates constant per node storage load.*

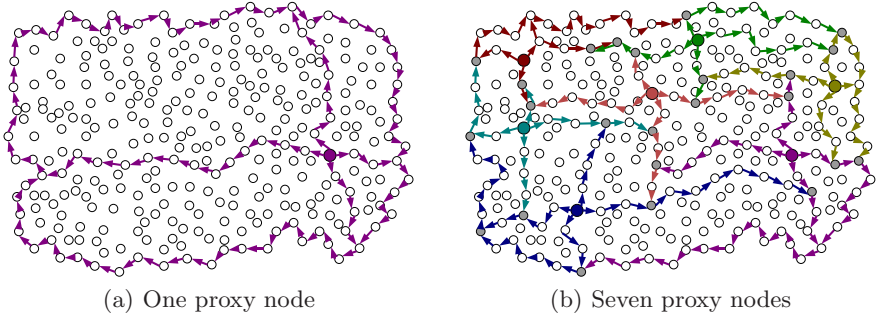


Fig. 2. Information mesh construction in an arbitrary sensor network

3.2 Arbitrary Sensor Networks

In an arbitrary sensor network, DSND can be implemented by using routing protocol GFG [1], which is known for its guaranteed packet delivery and has been used for quorum formation in the quorum-based location service [9, 8].

An arbitrary proxy node generates four registration messages carrying its location information respectively for the four directions, i.e., the north, the south, the west, and the east. Then it sends them to the corresponding directional foremost A-node neighbors. These registration messages are retransmitted by receiver nodes following protocol GFG. Specifically, upon receiving a registration message, an A-node retrieves the embedded node information from the message, stores the information locally and then greedily forwards the message to its foremost A-node neighbor in the same direction. When a registration message reaches a void area, it is switched to the *face routing* mode and then passed around the void area. Greedy forwarding resumes whenever possible.

If the source is the only proxy node in the network, a registration message will finally stop at the globally foremost A-node in its transmission direction, and its transmission path will include the entire network boundary, as shown in Fig. 2(a) where the registration paths of the only proxy node is highlighted by arrowed colorful lines. In the case that there is more than one proxy node in the network, proxy nodes' registration paths intersect each other inside the network and/or overlap on the network boundary. For two intersecting registration paths, they will be either in a *node-sharing situation* or in a *link-crossing situation*. In the former situation, the two paths intersect at a common node, while in the latter situation, they have a pair of crossing links. It can be observed that, a link-crossing intersection can be easily transformed to a node-sharing intersection in a localized way and without extra message transmission. Therefore, for any two different proxy nodes, their registration paths are guaranteed to have some A-node(s) in common. Then these common nodes apply the blocking rule as in the context of grid sensor networks. Finally, an information mesh structure is established as a result. Figure 2(b) shows an information mesh created by 7 proxy nodes in an arbitrary sensor network. In this figure, proxy nodes and their

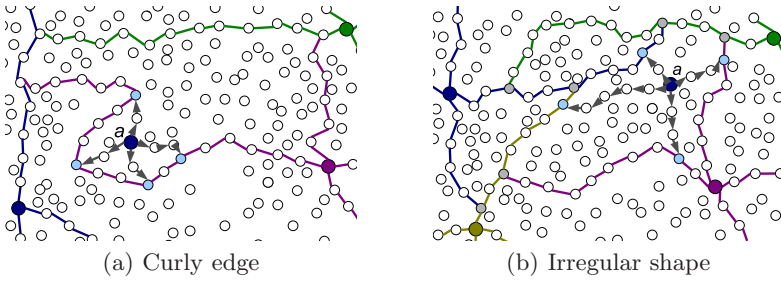


Fig. 3. Cross lookup in an arbitrary sensor network

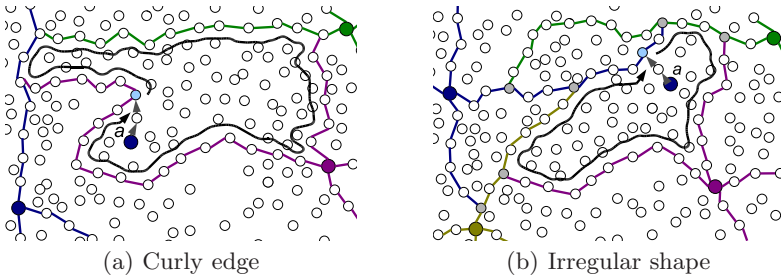


Fig. 4. Perimeter lookup in an arbitrary sensor network

registration paths are differentiated by different colors, and gray dots represent the nodes where the blocking rule applies.

When an A-node a wants to find its target proxy, it sends a query message to its directional foremost neighbors. In each direction, the message is retransmitted through protocol GFG and stops at the first receiver A-node that resides on the information mesh. Then this A-node sends a a positive reply containing its locally stored proxy node information. Node a selects its target proxy from positive replies. However, if there does not exist any proxy node in the network, which is possible when all the R-nodes become unavailable, such a query message will reach a boundary A-node b and then traverse the entire network boundary starting from there, by the property of GFG. In this case, once the query message gets back to b along the network boundary, b sends a a negative reply, indicating the failure of proxy lookup.

Under the assumption of $cR \geq 2sR$, if the sensor field is fully covered, no void area exists in the network topology, and the greedy forwarding part of GFG will never fail [13]. As a result, every cell in the information mesh has a rectangular shape, and the cross lookup method always works. However, if void areas (e.g., due to uncoverable obstacles) do exist in the network topology, messages are routed along the perimeters of the void areas, leading to zigzag message transmissions and thus possible cross lookup failure. Figure 3, where arrowed gray lines indicate query paths, shows two examples. In the first example, the query messages of A-node a all hit the same curly edge of its home cell; in the second

example, the home cell of a is composed of five edges, causing that no query message reaches the northmost edge. Apparently, a fails to find its true target proxy in these two cases.

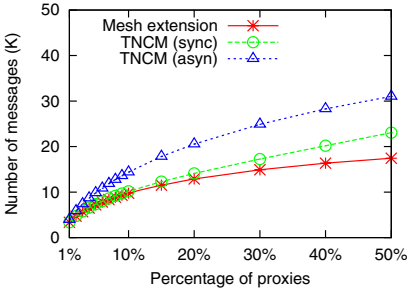
To ensure successful proxy node lookup in such undesired situations, an alternate *perimeter lookup* method can be used. By this method, the requesting A-node a sends a query message to an arbitrarily selected direction through GFG. The query message will hit a 's home cell perimeter at certain A-node, called *entry node*, which then retransmits the message along the cell perimeter. The query message picks up the information of the closest proxy node that it have seen during its perimeter traversal. After it travels all the way along the cell perimeter back to the entry node, it has found the target proxy of a . Therefore, upon receiving the query message back, the entry node immediately forwards the message back to a as a reply. This perimeter lookup method is illustrated in Fig. 4 where light blue dots denote the entry nodes that start perimeter traversal.

3.3 Performance Evaluation

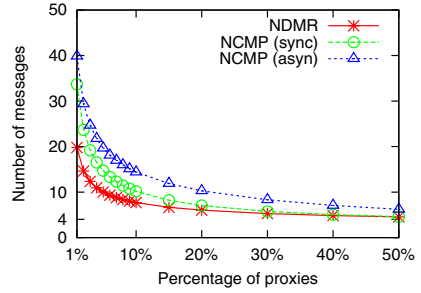
We simulate algorithm DSND in a large-scaled sensor network. The network consists of 10,000 nodes and fully covers the sensor field with average node density of 8 – 9. During our simulation, we run two sets of experiments. In the first set, the network is set to be a synchronous environment with simultaneous execution and unified link delay; in the second set, the network is configured to be an asynchronous environment where nodes start the protocol maximally 30 simulated time units off each other, and each communication link has transmission delay of 10 simulated time units at most. We choose 1%-, 2%-, \dots , 10%-, 20%-, \dots , and 50%-service-provider settings. For each setting, we execute DSND over 100 randomly generated network scenarios in order to get average experimental results, which are given in Fig. 5. Based on these results, we study the performance of DSND, as the Percentage of Proxies (PP) varies, using the following metrics:

- Total Number of Construction Messages (TNCM): the total number of messages transmitted in the network for information mesh construction;
- Number of Construction Messages per Proxy (NCMP): the average number of messages generated in the network by an arbitrary proxy node for information mesh construction;
- Number of Discovery Messages per Requester (NDMR): the average number of node discovery messages generated in the network by a requester node (reply messages are not counted);
- Target over Closest Ratio (TCR): the average ratio of the distance of a requester to its target proxy over its distance to the closest proxy node;
- Probability of $TCR = 1$ (PTCR1): the probability that the $TCR = 1$ for an arbitrary requester in the network.

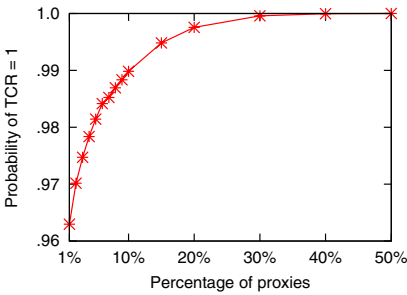
Figure 5(a) shows TNCM in relation with PP. For reference, mesh extension is also drawn in the figure. As PP grows, the information mesh has a more complex structure, therefore exhibiting an increasing mesh extension and a growing construction message overhead, which correspond to the ascending trend of the



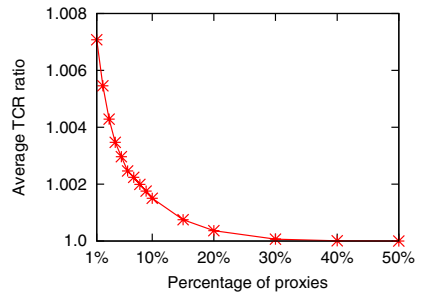
(a) Total number of construction messages



(b) Number of control messages per node



(c) Probability of TCR = 1



(d) Average TCR ratio

Fig. 5. Experimental results based on a sensor network of 10,000 nodes

curves in the figure. We can find that, in all the PP cases, TNCM in the asynchronous environment is always higher than that in the synchronous environment. This is because of the extra messages used for consistency maintenance. The results shown in this figure clearly indicates that DSND has considerably low message overhead (compared with the size of the network).

Figure 5(b) illustrates NDMR and NCMP in relation with PP. NDMR has only one corresponding curve since it is subject to mesh structure but not synchrony. From the figure, we can see that, as PP goes up, both NCMP and NDMR drop and approach 4. The reason is that, when the density of proxies increases, a requester’s node discovery message (or a proxy node’s registration message) travels a distance of decreased length (on average) in each direction before finding a proxy (resp., being blocked), and the travel distance can be as low as 1-hop, resulting in merely 4 node discovery messages (resp., registration messages) in the extreme case. It can also be observed that a proxy node uses slightly more construction messages in the asynchronous environment than in the synchronous environment due to the extra cost of consistency maintenance. The results shown in this figure again verify the low message overhead of DSND.

Figures 5(d) and 5(c) respectively show TCR and PTCR1 as a function of PP. The results in the two figures are regardless of the (synchronous or asynchronous) nature of the execution environment. From Fig. 5(d), we can see that PTCR1 is

always larger than 96%, and that it actually increases as PP goes up and closely approaches 1.0 after PP reaches 10%. From Fig. 5(d), we observe that, TCR is nearly equal to 1 in all the PP cases. This is because of the low probability of $TCR > 1$ (confirmed by Fig. 5(c)). We also observe that, the two curves decline and approach 1 closer and closer as PP increases. This phenomenon is due to the decreasing probability of $TCR > 1$ in case of increasing PP. The results in the two figures demonstrate the satisfactory distance performance of DSND.

4 The Mesh-Based Relocation Protocol

In this section, we are going to propose the *Mesh-based Sensor Relocation Protocol (MSRP)* to solve the sensor relocation problem defined in Sec. 1, based on algorithm DSND. For simplicity, we will present protocol framework only and leave implementation details for the next section.

At initiation, each R-node spontaneously takes the nearest A-node as proxy, by sending that A-node a delegation request. After being chosen as proxy, an A-node executes algorithm DSND to construct an information mesh. While being awake, a R-node monitors the liveness of its proxy; once it finds that its proxy fails, it moves to replace the proxy node directly. Upon an ordinary (i.e., non-proxy) A-node failure, the A-nodes neighboring the failed A-node cooperate to discover a *replacement*, which is defined as the nearest delegated R-node of the target proxy of the failed A-node, by using DSND. For brevity, the target proxy of a failed A-node is referred to as *replacement proxy*. During the replacement discovery process, the two lookup methods, i.e., cross lookup and perimeter lookup, may be selectively used, depending on requirement. For the cross lookup method, the northmost, the southmost, the eastmost and the westmost neighbors of the failed A-node, as *servers*, send four query messages respectively to the north, the south, the east, and the west direction. After getting replies, they exchange their discovery results through underlying routing protocol to find the replacement proxy. For the perimeter lookup method, only one neighbor, say the northmost neighbor, acts as the server of the failed A-node. It sends a query message to an arbitrarily selected direction, and later receives a reply that contains the replacement proxy's location. Whichever lookup method is employed, only the server that is closest to the replacement proxy is considered as *replacement discoverer*.

The replacement discoverer issues a relocation request to the replacement proxy, which then grants the request by sending back an ACK message to the discoverer. After receiving the ACK message, the replacement discoverer starts a relocation path discovery process by sending an action message to the replacement proxy through routing protocol GFG [1]. Relocation path discovery is in essence a QoS routing process. Its objective is to establish a path, between a replacement discoverer and a replacement proxy, which yields minimized energy usage and time delay for shifted relocation. Recall that the shifted relocation method requires all the node along a relocation path to shift their position toward a failed A-node. From energy-saving point of view, node relocation should involve shortest total moving distance and a least number of moves. In other

words, a relocation path is expected to have both minimized path length and minimized hop count. On the other hand, to reduce relocation latency in the case of simultaneous shifting, longest hop length in a relocation path must be minimized, which is virtually equivalent to maximizing the hop count of the relocation path. Under this contradictory circumstance, MSRP takes COST over PROGRESS ratio [10] as routing criterion and combine it with the greedy forwarding part of GFG. Here, COST is defined as the Euclidean distance from current node to the considered next hop, and PROGRESS is defined as the difference between the Euclidean distance from current node to the destination, i.e., a replacement proxy, and the Euclidean distance from the considered next hop to the destination. More formally, denote by a_0 the source, i.e., a replacement discoverer, and by a_i the i -th hop along the path from a_0 to destination d . Then, the $(i + 1)$ -th hop a_{i+1} must be closer to d than a_i and meanwhile minimize the following objective function:

$$f(a_{i+1}) = \frac{|a_i a_{i+1}|}{|a_i d| - |a_{i+1} d|} \quad .$$

Normally, the shifted node relocation process starts after a relocation path is constructed. However, in order to save messages and reduce relocation latency, in MSRP, node relocation and relocation path discovery are performed in parallel. More specifically, after sending the action message, the discoverer moves to the failed node's location right away; intermediate nodes move to the position of their priori hop after forwarding the action message. As for the replacement proxy, after receiving the action message, it first informs the replacement node to fill its current position and then itself moves toward the location of its prior hop. In order not to jeopardize the information mesh or the execution of other network protocols, every relocating node transfers all its local data to the newcomer at its original position immediately after.

5 Implementation Details

A straightforward implementation of the basic protocol design is not sufficient in practice. Some mechanisms must be provided to deal with impact factors such as node failures and node contention. Because of page limitation, we will discuss these implementation issues at very short length. Details can be found in [7].

The storage load of an edge intersection node a in the information mesh is bounded by the number of incidental mesh edges of the node, which may however not be constant and can be equal to the degree of the node in the worst case. In order to keep constant per node storage load, MSRP requires an edge intersection node store data pointer (nearly at no cost) instead of actual data. In addition, unlimited R-node delegation could also cause inconstant storage load, which may be as bad as $O(n')$, per A-node in the case that R-nodes are densely distributed. To handle this problem, a limit can be put on the number of an A-node's accepted delegation requests.

Although protocol MSRP is designed to deal with node failures, its execution is not automatically fault-tolerant. The impacts from node failures on MSRP can include loss of proxy information, loss of control messages and replacement failure. In order to make its execution resilient against run-time node failures, MSRP employs the following fault-tolerance techniques: thick message transmissions, transmission retrials, and real-time monitoring and taking over.

Because of the distributed nature of MSRP, node contention is very likely to happen during node relocation processes. There are two types of node contention. Type-I is that multiple A-nodes are attempting to relocate the same R-node to replace different failed A-nodes; type-II is that an A-node appears in multiple relocation paths and is required to shift its position along those paths. MSRP handles the two types of node contention on a first-come-first-serve basis.

Acknowledgments. This research is partially supported by the UK Royal Society Wolfson Research Merit Award, NSERC Collaborative Research and Development grant CRDPJ 319848-04 and NSERC Strategic Grant STPGP 336406.

References

1. Bose, P., Morin, P., Stojmenovic, I., Urrutia, J.: Routing with Guaranteed Delivery in Ad Hoc Wireless Networks. In: Proc. of ACM DIALM, pp. 48–55 (1999)
2. Frey, H., Stojmenovic, I.: On Delivery Guarantees of Face and Combined Greedy-Face Routing Algorithms in Ad Hoc and Sensor Networks. In: Proc. of ACM MobiCom, pp. 390–401 (2006)
3. Gallais, A., Carle, J., Simplot-Ryl, D., Stojmenovic, I.: Localized Sensor Area Coverage with Low Communication Overhead. In: Proc. of IEEE PerCom, pp. 328–337 (2006)
4. Heo, N., Varshney, P.K.: Energy-Efficient Deployment of Intelligent Mobile Sensor Networks. *IEEE Tran. on Systems, Man, and CyberNetics - Part A: Systems and Humans* 35(1), 78–92 (2005)
5. Li, X., Santoro, N.: ZONER: A ZONE-based Sensor Relocation Protocol for Mobile Sensor Networks. In: Proc. of IEEE LCN/WLN, pp. 923–930. IEEE Computer Society Press, Los Alamitos (2006)
6. Li, X., Santoro, N.: An Integrated Self-Deployment and Coverage Maintenance Scheme for Mobile Sensor Networks. In: Cao, J., Stojmenovic, I., Jia, X., Das, S.K. (eds.) *MSN 2006*. LNCS, vol. 4325, pp. 847–860. Springer, Heidelberg (2006)
7. Li, X., Santoro, N., Stojmenovic, I.: Mesh-based Sensor Relocation for Coverage Maintenance in Mobile Sensor Networks (Full Version), <http://www.scs.carleton.ca/~xlii/MSRP.pdf>
8. Liu, D., Stojmenovic, I., Jia, X.: A scalable quorum based location service in ad hoc and sensor networks. In: Proc. of IEEE MASS, pp. 489–492. IEEE Computer Society Press, Los Alamitos (2006)
9. Stojmenovic, I.: A scalable quorum based location update scheme for routing in ad hoc wireless networks. Technical Report, TR-99-09. SITE, Univ. of Ottawa (1999)
10. Stojmenovic, I.: Localized network layer protocols in wireless sensor networks based on optimizing cost over progress ratio. *IEEE Network* 20(1), 21–27 (2006)

11. Wang, G., Cao, G., Porta, T.L.: Proxy-Based Sensor Deployment for Mobile Sensor Networks. In: Proc. of IEEE MASS, pp. 493–502. IEEE Computer Society Press, Los Alamitos (2004)
12. Wang, G., Cao, G., Porta, T.L., Zhang, W.: Sensor Relocation in Mobile Sensor Networks. In: Proc. of IEEE INFOCOM, pp. 2302–2312. IEEE Computer Society Press, Los Alamitos (2005)
13. Xing, G., Lu, C., Pless, R., Huang, Q.: Impact of sensing coverage on greedy geographic routing algorithms. IEEE Tran. on Parallel and Distributed Systems 17(4), 348–360 (2006)