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

Xu Li
SCS, Carleton University, Canada
Email: xlii@connect.carleton.ca

Nicola Santoro
SCS, Carleton University, Canada
Email: santoro@scs.carleton.ca

Ivan Stojmenovic
SITE, University of Ottawa, Canada
Email: ivan@site.uottawa.ca

***Abstract*— Sensor relocation protocols can be employed as a fault-tolerance approach to reduce or complement coverage loss caused by node failures. In this paper, we introduce a novel localized structure, *information mesh*, for publishing and retrieving distance-sensitive data like location information. Based on the concept of information mesh, we then propose a Mesh-based Sensor Relocation Protocol (MSRP) for mobile sensor networks. The proposed protocol maintains a sensor network's overall sensing coverage by replacing failed sensors with nearby redundant ones using minimized time delay and balanced energy consumption. We show that MSRP is superior to the existing relocation protocols due to its localized message transmissions, optimal (constant) per node storage load, and its guaranteed nearby replacement node discovery and node replacing.**

I. Introduction

Mobile sensor networks, as a new paradigm of wireless sensor networks, are known for their particularity, node mobility. Because mobile nodes are able to take intelligent physical actions like escaping from dangerous situations or responding to interesting events by executing sophisticated protocols, mobile sensor networks are more flexible and adaptive to unknown or hazardous environments than static wireless sensor networks. Recently, many unique research issues are emerging in mobile sensor networks. One of them is maintaining sensing coverage through autonomous node movement [9].

A. Motivation

Sensing coverage (or coverage for short) is an important QoS factor in sensor networks. 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. As a sensor network operates, its coverage decreases because of node failures. The reasons why nodes fail are multifold. For example, a node may run out of battery power and stop functioning at any time, and it may suddenly die because of hardware defects or due to harsh environment conditions like extreme temperature. In these cases, to maintain quality of service, a sensor network must have the capability of preserving its coverage in the presence of node failures.

In static sensor networks, using a large number of redundant nodes is the only way to tolerate node failures. Relevant research concentrates mainly on how to schedule sensor activity to save energy without jeopardizing network coverage [17], [3]. However, in mobile sensor networks, node mobility can be exploited to facilitate coverage maintenance. That is, deploy a moderate number of redundant sensors and strategically relocate them as needed to fill the position of failed nodes. This type of movement-assisted coverage maintenance approaches are called *sensor relocation*.

To our knowledge, only three sensor relocation protocols WCP [14], WCPZ [16], and ZONER [10] were proposed for mobile sensor networks in the literature. They are all inferior for possible applications, compared to the protocol proposed in this article, for variety of reasons. All of them rely on global/cross-network message transmissions for discovering nearby replacement sensors, generating $O(n'\sqrt{n})$ messages, where n' and n are respectively the number of redundant sensors and the number of non-redundant sensors. They all require non-constant storage load $O(n')$. Further, WCPZ depends on the assumption of the preknowledge of the border of the sensor field, and WCP has non-constant delay and unbalanced energy usage. Both WCP and WCPZ do not address the issue of guaranteed discovery of a replacement sensor when one in fact exists and is connected to the area where it could move. A brief description and a comparative analysis on these three relocation protocols can be found respectively in Sec. II and in Sec. VI.

B. Problem statement

We consider a connected mobile sensor network deployed in an unbounded 2-D 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 [5], [20], [15], [4], [18] after its initial placement. The nodes that constitute the network are called *active nodes (or A-node)*. A-nodes always remain active and participate in all kinds of network operations. We also assume that some predefined *redundant nodes (or R-node)* 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. Their communication radius cR is at least twice as large as their sensing radius sR. Every node is aware of its own geographical location (expressed as a (x,y) coordinate), and may fail at any time for any reason.

Our research goal is to develop, based on above network model, a sensor relocation protocol that can maintain a network's coverage by relocating nearby R-nodes to the position

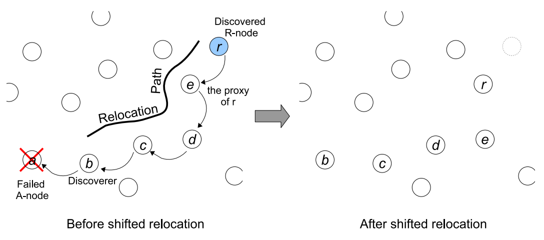


Fig. 1. An illustration of shifted node relocation

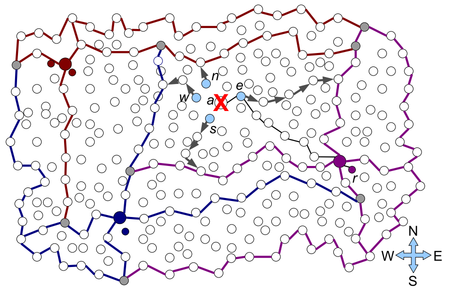


Fig. 2. A general view about how MSRP works

of failed A-nodes. Note that, in a disconnected network, there is no guarantee that failed A-nodes are successfully replaced. In this case, we additionally assume the network alway remains connected in spite of node failures. Furthermore, 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.

C. Our contributions

In this paper, we propose a mesh-based sensor relocation protocol (MSRP) to solve the sensor relocation problem defined in Sec. I-B. With zero preknowledge of the sensor field, MSRP accomplishes the following two tasks: moving a redundant sensor to replace a failed one (*node relocation task*) and discovering a redundant sensor for sensor replacement (*replacement discovery task*).

Protocol MSRP fulfills the node relocation task by a shifted node relocation method. That is, establish a path between a failed A-node and a R-node in a localized way, and shift all the nodes' position along the path toward the failed A-node. This method uses a localized relocation path discovery mechanism, and generates constant relocation delay (bounded by cR) and balanced energy consumption. A shifted node relocation process is illustrated in Fig. 1, where an arrow from a node points to the target location of the node. Two variants of the shifted node relocation method been presented in the literature [16], [10]. But, they both have weakness in their relocation path discovery part, when compared with our protocol MSRP.

Protocol MSRP accomplishes the replacement discovery task through a sub-algorithm, Distance-Sensitive Node Discovery algorithm (DSND), which provides nearby node discovery guarantees. By DSND, some A-nodes are selected by R-nodes as proxy and construct an *information mesh* over the network in a localized way. The more proxy nodes, the more localized the construction. This information mesh distributedly stores the location information of all the proxy nodes with optimal (constant) per node storage load $O(1)$. Upon an A-node failures, the A-node neighbors of the failed node find a nearby proxy node via the information mesh and take the proxy node's nearest delegated R-node as the failed node's replacement. The message complexity of algorithm DSND is never larger than that of the node discovery methods employed by the existing relocation protocols.

Figure 2 gives a general view about how protocol MSRP works. In this figure, R-nodes are represented by small colorful dots; proxy nodes are denoted by the big dots in the color corresponding to their delegated R-nodes'; the information mesh built by proxy nodes is highlighted through colorful links. After an A-node a fails, its northmost, southmost, westmost, and eastmost A-node neighbors, i.e., n, s, w, and e, work in collaboration to find a replacement, R-node r in this example, in a's vicinity. The paths along which the four nodes discover r are shown by arrowed gray lines. The replacement node r is then shiftedly relocated to fill the position of a along the path indicated by thin black lines.

To sum up, our proposed protocol MSRP is a localized preknowledge-independent algorithm. It guarantees replacement node discovery and node replacing, using optimal per node storage load $O(1)$, constant relocation delay and balanced energy consumption. It outstands all the existing sensor relocation protocols [14], [16], [10], especially in the scenarios where there are large numbers of randomly scattered R-nodes.

D. Paper outline

The remainder of the paper is organized as follows: Section II summarizes some related work in the literature; Section III introduces the concept of information mesh and presents the Distance-Sensitive Node Discovery algorithm (DSND); Section IV proposes the Mesh-based Relocation Protocol (MSRP) on the basis of DSND; Section V discusses the implementation details of MSRP; Section VI analyzes the characteristics of MSRP in comparison with the three existing relocation protocols; Section VII concludes the paper and indicates our future work.

II. Related Work

In this section, we are going to first review the Greedy-Face-Greedy routing protocol [1] and the quorum-based location service [12], [11], and then we describe the three existing sensor relocation protocols [14], [16], [10].

A. Greedy-Face-Greedy routing

Bose, Morin, Stojmenovic and Urrutia proposed a Greed-Face-Greedy routing protocol (GFG) for wireless sensor networks [1]. The GFG is a combination of a simple greedy forwarding strategy and the face routing technique. It is a stateless routing protocol in the sense that nodes do not need to remember any routing information such as routing table or route list. The GFG is the first localized protocol that provides guaranteed packet delivery. There exist several other combined Greedy-Face routing protocols such as GOAFR+ [7] and GPVFR [8], but they are in essence variants of GFG.

In a GFG routing process, greedy forwarding is applied whenever possible, while face routing is used only for passing packets around the void areas (or, dead-ends) that block greedy forwarding. A node greedily forwards a packet toward the destination by choosing as the next hop its neighbor closest to the destination. In the case that the current node X does not have neighbor closer to the destination than itself, the packet is forwarded in face routing mode using right-hand/left-hand rule until the destination or a node closer to the destination than X is found. Face routing works for arbitrary planar graphs. GFG uses Gabriel Graph (GG), where the diametral disc of each edge contains no other vertices than the two edge ends, as planar graph to support face routing. GG does not require any message to be exchanged between neighbors if each node is aware of geographic positions of itself and its neighbors. The right-hand (left-hand) rule for traversing a face is that a packet is forwarded in the clockwise (resp., counterclockwise) direction along the perimeter of the face. As proven in [2], when GG is applied, greedy forwarding recovery in face mode is guaranteed when traversing the first face.

B. Quorum-based location service

Stojmenovic proposed the quorum-based location service to support geographic routing in ad hoc networks [12], [11]. By this service, each node, when necessary, forwards its current position to all the nodes located in a “column” of certain thickness. That is, it sends its location in both north and south direction to reach the north and south boundaries of the network. When a source node wants to communicate with a destination node, it has to search for the location of the destination if its local record about the destination is out of date. First, the source queries its q-hop neighborhood for the destination’s location. If the answer is negative, or if the obtained information is not fresh enough, the search continues in the east and west direction with certain thickness. One more request may be sent directly to the destination to take the advantage of the possible correctness of the best information obtained during the q-hop neighborhood search. The three searches are performed independently. The trace of the eastbound and westbound search form a row, which intersects the columns of all the other nodes, including that of the destination. As the query message travels along the row, it picks the latest location information about the destination. When the message reaches the ends of the row, the message will be forwarded to the destination, which then replies directly with correct location and possibly form a route for future data transmission. Alternatively, intersection nodes may reply immediately if the information is sufficiently fresh.

The main disadvantages of this quorum-based location service are that location update still has to cross the entire network, and that, in the case that all the nodes are collinear, every node may have to store every other node’s location, resulting in non-constant per node storage load.

C. Sensor relocation

Wang, Cao and Porta presented a proxy-based sensor relocation protocol (referred to as WCP) for the sensor networks composed of both static nodes and mobiles [14]. By WCP, static nodes locally broadcast their locations and identities to construct a Voronoi diagram. Mobile nodes periodically broadcast within certain predefined radius their location information and base prices (initially set to zero) as service advertisement. Based on received service advertisements, a static node create a service provider list. Once a static node finds a coverage hole within its Voronoi polygon, it estimates the hole size and tries to bid a closest mobile node with lowest base price in its service provider list. In the case that a mobile node receives multiple bidding messages, it is bid by the message with largest hole size and then moves to fill the corresponding hole. Hence, mobile nodes intend to move to large holes from small ones, and stay still only when no larger holes can be detected. To save energy, a mobile node logically moves to its target location by choosing proxies; it performs actual movement when its target location is the final location.

Wang, Cao, Porta and Zhang presented a grid-quorum-based relocation protocol (referred to as WCPZ) for mobile sensor networks [16]. This protocol employs the quorum-based location service [12], [11], in modified forms, to find replacement for failed sensors. By WCPZ, the network field is partitioned into a 2-D grids. In each grid, one node is elected as grid head and takes the responsibility to collect the location of all the grid members. Based on grid members’ location, a grid head determines redundant grid members and detects sensing holds. A row of grids is called supply quorum, while a grid column is called demand quorum. Each grid head publishes the information about the redundant nodes inside its grid to all the grid heads in its residing supply quorum. When a grid head detects a sensing hole, it broadcasts a request within its residing demand quorum to discovery the closest redundant node. Because every demand quorum intersects with all the supply quorums, a redundant node can always be found (if any exists). WCPZ uses restricted flooding to find a satisfactory relocation path and relocate the discovered redundant node along the path in a cascaded (shifted) way.

Li and Santoro presented a zone-based relocation protocol (ZONER) for mobile sensor networks with previously deployed redundant sensors [10]. This protocol is also a variant of the quorum-based location service. Each redundant node register itself with all the non-redundant nodes within a vertical registration zone. After a non-redundant node failed, its westmost neighbor and eastmost neighbor initiate a node discovery process in their bounded horizontal request zones. Because the request zones intersects with a number of registration zones, the non-redundant nodes in the intersection areas can reply with the requested information. Then the discovered redundant node with shortest relocation path is relocated, in a shifted manner, to replace the failed node. Although ZONER and WCPZ [16] have similarity in their node discovery and node relocation methods, they differ a lot from each other in that ZONER requires no preknowledge of the sensor field and guarantees replacement discovery (by resorting to face routing) and node replacing.

To our knowledge, above three sensor relocation protocols are the only ones that were proposed for the purpose of coverage maintenance in the literature. WCP [14] is a flooding-based protocol with direct relocation method that can generate non-constant relocation delay. WCPZ [16] requires preknowledge of the border of the network and may fail in the case that there exist void areas in the network. Due to the application of the quorum technique [12], [11], both WCPZ and ZONER [10] has message complexity $O(n'\sqrt{n})$ for replacement discovery and generate non-constant per node storage load $O(n')$, where n' and n are respectively the number of redundant sensors and the number of non-redundant sensors. A more detailed analysis on the characteristics of the three protocols is given later in Sec. VI. Through study, we can find that the existing sensor relocation schemes all have major drawbacks and are thereby inferior for possible applications.

III. Distance-Sensitive Node Discovery

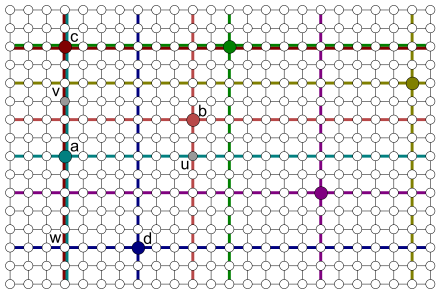
In this section, we devise a localized *Distance-Sensitive Node Discovery algorithm (DSND)* based on the network model described in Sec. I-B. This algorithm will be employed by the *Mesh-based Sensor Relocation Protocol* (to be proposed later, in Sec. IV) for replacement discovery.

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 multiple R-nodes are allowed to share a common proxy. Proxy nodes record the location of their delegated R-nodes in their local repositories and together construct an information mesh over the network. This information mesh distributedly stores the location information of all the proxy nodes. When a nearby R-node is wanted, an A-node just need to find a proxy node in its vicinity.

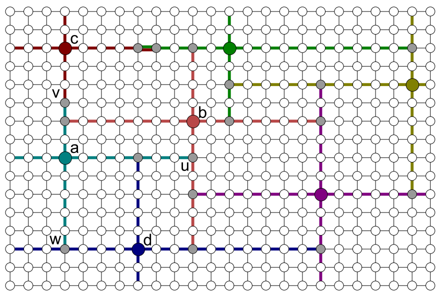
By above description, the core of DSND consists of two parts: *information mesh construction* and *proxy node lookup*. For easy understanding, in the following, we present the two key components first in well-structured grid networks and then in arbitrary network scenarios, ignoring all the practical impact factors and implementation details.

A. Grid sensor networks

In a grid sensor network, A-nodes are placed exactly at the intersection points of a grid structure. Each boundary node has either two or three neighbors, while every internal node has four neighbors that are respectively located in its north side, south side, west side, and its east side. In this case, any A-node is able to find out its own role in the grid structure simply by counting the number of its neighboring A-nodes.



(a) A complete mesh structure



(b) A (pruned) information mesh

Fig. 3. Information mesh construction in a grid sensor network

We denote such a grid sensor network by $G(A, R)$ (or simply by G), and the number of proxy nodes in G by $\nu(G)$. The two notations A and R represent the set of A-nodes and the set of R-nodes in G, respectively. When G is given, $\nu(G)$ can be written as ν without ambiguity. By the definition of proxy node selection, $\nu \leq Min\{n, n'\}$ where $n = |A|$ and $n' = |R|$.

1) Constructing information mesh: Consider only the residing rows and columns of the proxy nodes in G. They intersect one another and form a mesh structure, as illustrated in Fig. 3(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, this mesh structure distributedly stores the location information of all the proxy nodes and therefor can be used for the purpose of proxy node lookup.

Let us examine the mesh structure shown in Fig. 3(a). Proxy node c is closer to the area above the mid-point A-node 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. By this observation, we define a blocking rule.

Definition 1 (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 $(|ua| > |ub|) \vee (|ua| = |ub| \wedge cline(a,b)) \vee (|ua| = |ub| \wedge \neg cline(a,b) \wedge north(a,b))$, where $cline(a,b)$ and $north(a,b)$ denote the case that a and b are (vertically or horizontally) collinear and the case that a is located in the north of b, respectively. And, when this blocking happens, we say "b blocks a at u".

The application of the blocking rule can lead to the merge of adjacent mesh cells and result in a pruned mesh structure, i.e., *information mesh*. We denote the information mesh constructed on top of G by $\mathcal{IM}(G)$ (or simply by $\mathcal{IM}$). Figure 3(b), where gray dots represent the A-nodes at which the blocking rule actually applies, shows the information mesh corresponding to the complete mesh structure in Fig. 3(a).

Definition 2 (Extension): The extension $\eta(\mathcal{IM})$ (or η for brevity) of information mesh $\mathcal{IM}$ is the length summation of all the edges of $\mathcal{IM}$.

Definition 3 (Home Cell): The home cell(s) of a A-node is the mesh cell where the A-node is located in or the mesh cells which it is adjacent by.

Definition 4 (SPV): The Set of Proxies in Vicinity (SPV) of an A-node is the set of proxy nodes whose residing grid rows/columns form the home cell(s) of the A-node.

Definition 5 (Target Proxy): The target proxy of an A-node is the nearest proxy node in the A-node's SPV.

***Lemma** 1:* The message complexity of information mesh construction is $O(\eta)$.

Proof Sketch: Observe that the edges in an information mesh $\mathcal{IM}$ are exactly the paths which proxy node location information travels along, and that, on each communication link in these edges, no more than two messages are transmitted. By this observation, the number of messages for constructing $\mathcal{IM}$ is bounded by $O(\eta)$. Hence, the lemma holds. ■

***Lemma** 2:* In a square grid network, $\eta \in O(\nu\sqrt{n})$.

Proof Sketch: Consider a complete mesh structure established without applying the block rule. The extension of this structure is bounded below $O(\nu\sqrt{n})$. Since an information mesh is the result of removing (by the blocking rule) some edges from such a complete mesh, its extension can not exceed $O(\nu\sqrt{n})$. Hence the lemma holds. Note that, this upper bounder is achievable in terms of order of magnitude, for example, when proxy nodes are all located on the same line along either the X axis or the Y axis. ■

***Lemma** 3:* In a square grid network, η can be as small as $O(\sqrt{\nu n})$

Proof Sketch: If, by any chance, the information mesh has a square grid structure with the same border as the network, then $\eta = \sqrt{\nu n}$. This proves the lemma. ■

Let us examine the grid sensor network in Fig. 4, where solid thick black lines form a Voronoi diagram of the proxy nodes. Consider an A-node a, whose nearest proxy node is c, in a shadowed triangle area in the figure. We can easily find that a's home cell perimeter does not include or partially include the residing grid row or column of c. This example

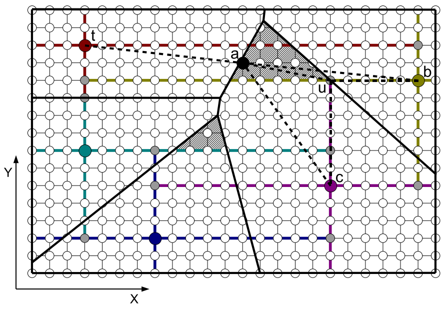


Fig. 4. An illustration of non-closest target proxy

indicates that, there is no guarantee that the target proxy of an A-node is the proxy node globally closest to the A-node.

***Lemma** 4:* Denote by t the target proxy of an A-node a and by c the proxy node closest to a. And, $t \neq c$. Let b be the proxy node in a's SPV, whose residing grid row (or column) passes through between a and c and intersects the residing grid column (resp., row) of c at an A-node u. Then, $|ub| \leq |uc|$.

Proof Sketch: Since $t \neq c$, there must be a blocking chain of length $k (k \geq 1)$ with the following format: $c \leftarrow p_1 \leftarrow, \cdots, \leftarrow p_k$, which means that, a proxy node p_1 blocks c, and a proxy node p_2 blocks p_1, $\cdots$, and a proxy node p_k blocks p_{k-1}. Assume that this chain of blocking happens along the Y axis. Then p_k could be either b or a proxy node located not closer, in X-direction, to c than b. Figure 4 shows the simplest case of this blocking chain, where $k = 1$ and $p_k = b$. Let us denote c by p_0 and consider two consecutive proxy nodes p_i and p_{i-1} $(1 \leq i \leq k)$ in the blocking chain. We have $|x_i - x_{i-1}| \leq |y_i - y_{i-1}|$, where (x_i, y_i) and (x_{i-1}, y_{i-1}) are respectively the coordinates of p_i and p_{i-1}. It is because that p_i, otherwise, can not block p_{i-1} in Y-direction. Therefore, $|x_k - x_{k-1}| = |\sum_{i=1}^{k}(x_i - x_{i-1})| \leq \sum_{i=1}^{k}|x_i - x_{i-1}| \leq \sum_{i=1}^{k}|y_i - y_{i-1}| = |\sum_{i=1}^{k}(y_i - y_{i-1})| = |y_k - y_{k-1}|$. This inequality indicates that, the distance between p_k and c in X-direction is not larger than their distance in Y-direction. Hence the lemma holds. ■

***Theorem** 1:* **In a grid network G, the Euclidean distance from an A-node a to its target proxy t is at most twice as long as the Euclidean distance between a and its globally nearest proxy node c.**

Proof Sketch: The theorem is valid if $c = t$. Suppose, otherwise, $c \neq t$, as shown in Fig. 4. By Lemma 4, $|bu| \leq |cu|$. Observe that angle $\angle cua$ can not be acute in any case. Thus ca is the longest side in triangle Δcua. Namely, $|cu| < |ca|$ and $|ua| < |ca|$. Then $|at| \leq |ab| \leq |bu| + |ua| \leq |cu| + |ua| < |ca| + |ca| = 2|ca|$. This proves the theorem. ■

For an A-node where the blocking rule applies, it does not store the location information that it blocks but adds a mark (nearly at no extra storage cost) to the A-node neighbor from which it receives the blocked information, such that it can later find the blocked information without actually storing it.

***Lemma** 5:* An information mesh has constant per node storage load $O(1)$.

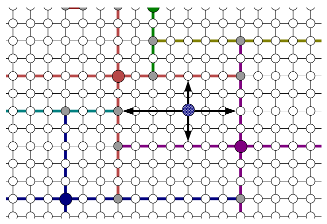
Proof Sketch: Each of the A-nodes that constitute the information mesh records at most one proxy node's location due to the application of the blocking rule. As for the nodes not part of the information mesh, they do not store any data about the information mesh at all. Hence, the lemma holds. ■

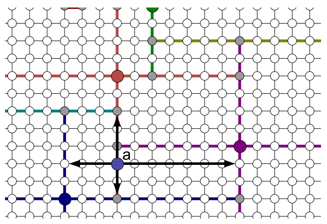
2) Discovering proxy nodes: The objective of proxy lookup is to identify the location of the target proxy of a requesting A-node. With the assistance of previously constructed information mesh, proxy lookup becomes fairly easy. Consider an A-node a in a cell of the information mesh. When it wants to find its target proxy node, it just inquires the A-nodes along its residing row and column in the grid structure in four directions, as shown in Fig. 5(a). 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 node simply through a local comparison. If there does not exist any proxy node in the network, proxy lookup will fail. A requesting A-node can be aware of such a proxy lookup failure after it reaches the border of the network in each of its query direction. Because the query paths of a requesting A-node form a cross, this type of proxy lookup method is called *cross lookup*.

Cross lookup can also be applied to the situation that a requesting A-node a is residing on the information mesh. In this case, a inquires along its residing mesh edges and stop at the farthest corners of its home cells on these mesh edges. By this means, it can reach all the mesh edges of its home cells and make right decisions. An example is given in Fig. 5(b), where query paths are highlighted by arrowed black lines. In this example, the requesting A-node a inquires along the common edge of its west-side home cell and its east-side home cell toward both the north and the south direction, passing through the northwest corner of its east-side home cell and gets to the northeast corner of its west-side home cell.

***Lemma** 6:* In a square grid network G, the message complexity of cross lookup is bounded by $O(\sqrt{n})$.

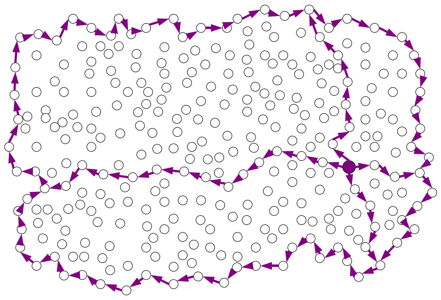
Proof Sketch: A cross lookup process is restricted within a search cell, which can be single mesh cell or a big cell composed of several mesh cells. In worst case, for example, when all the proxy nodes are located on the same border of the network, a search cell spans the entire network, and a requesting A-node in the search cell will inquire all the way along its residing grid row and/or column, generating $O(\sqrt{n})$ messages. Hence, the lemma holds. ■


(a) In-cell case



(b) On-edge case

Fig. 5. Cross lookup in a grid sensor network


(a) One proxy node

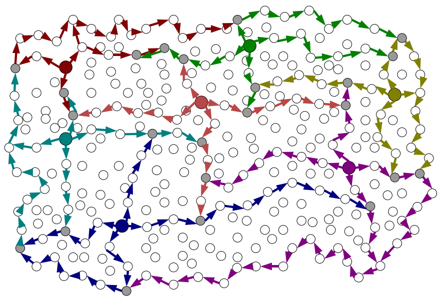

(b) Seven proxy nodes

Fig. 6. Information mesh construction in an arbitrary sensor network

***Theorem** 2:* **In a square grid network G, the message complexity of algorithm DSND is $O(\psi(G)\sqrt{n})$, where $\psi(G)$ is not larger than ν and can be as small as $\sqrt{\nu}$.**

Proof Sketch: It follows from Lemma 1-3 and 6. ■

B. Arbitrary sensor networks

In an arbitrary sensor network, there is no grid structure that we can make use of for information mesh construction and proxy node lookup. Under this circumstance, we accomplish our goal by using routing protocol GFG [1], which is known for its guaranteed packet delivery and has been used to form quorum in the quorum-based location service [12], [11].

1) Constructing information mesh: For an arbitrary proxy node, it 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, namely, the northbound message to the northmost A-node neighbor, and the southbound message to the southmost A-node neighbor, and so on. These registration messages are retransmitted by receiver nodes following protocol GFG. More specifically, upon receiving a registration message, an A-node retrieves the embedded node information from the message, stores it in the local storage, records the message's designated transmission direction 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 in the clockwise (counterclockwise) direction by the left (resp., right) hand rule. Greedy forwarding resumes whenever possible.

If the source is the only proxy node in the network, due to the absence of the network's boundary information and the nature of GFG, 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. 6(a) where the registration paths (i.e., the transmission paths of the registration messages) 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 one and another 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 case, the two path intersect at a common node, while in the latter case, they have a pair of crossing links. A link-crossing intersection can be easily transformed to a node-sharing intersection in a localized way without extra message transmission, as explained in Appendix.

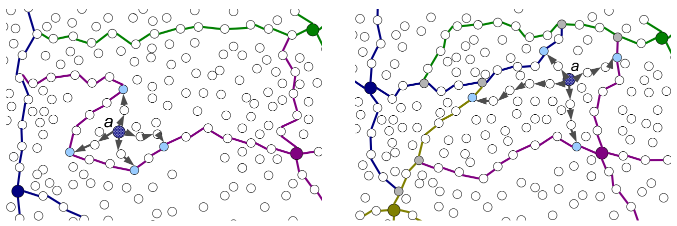
By above analysis, for any two different proxy nodes, their registration paths are guaranteed to have some A-node(s) in common. Then these common nodes apply the blocking rule as in the context of grid sensor networks. Finally, an information mesh structure is established as a result. Figure 6(b) shows an information mesh created by 7 proxy nodes in an arbitrary sensor network. In this figure, proxy nodes and their registration paths are differentiated by different colors, and gray dots represent the nodes where the blocking rule applies. Because of the straightforward implementation of the blocking rule, Lemma 5 holds also in arbitrary network scenarios.

***Theorem** 3:* **In an arbitrary sensor network, an information mesh has constant per node storage load $O(1)$.**

2) Discovering proxy nodes: The implementation of cross lookup is simple. A requesting A-node a sends a query message to its directional foremost neighbors. Each of these messages 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. 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 rely, indicating the failure of proxy lookup. For the requesting A-node a, if all the replies it receives are positive, it can easily determine its target proxy node; if at least one of them is negative, it knows that its proxy lookup fails.

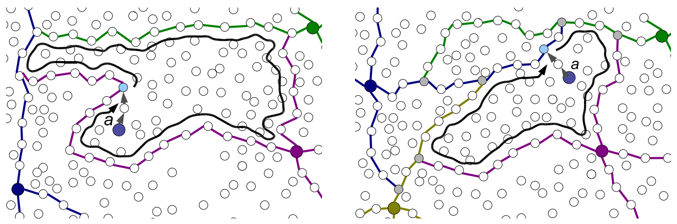
Under the assumption of $cR \geq sR$ (see Sec. I-B), if the entire sensor field is fully covered, as proven in [19], no void area is going to appear in the network topology, and the greedy forwarding part of GFG will never fail. As a result, every cell in the information mesh has a rectangular shape, and the cross lookup method always works. However, it may not be the case in reality due to the complex geographic feature of the sensor field. If uncoverable obstacles such as hills and lakes present in the sensor field, void areas can appear in the network topology. Under this circumstance, messages are routed along the perimeters of the void areas, causing zigzag message transmissions and thus the failure of the cross lookup. Figure 7, where arrowed gray lines indicate request paths, shows two examples. In the scenario demonstrated by Fig. 7(a), the query messages of A-node a all hit the same curly edge of its home cell; in the scenario illustrated by Fig. 7(b), 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 node 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, a requesting A-node a sends a query message to an arbitrarily selected direction through GFG. The query message will hit node a's home cell perimeter at certain A-node, called *entry node*, which then retransmits the message along the cell perimeter, e.g., in the clockwise direction. 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 the requesting A-node. Therefore, upon receiving the query message back, the entry node immediately forwards the message back to the requesting A-node a as a reply. This perimeter lookup method is illustrated in Fig. 8 where light blue dots denote the entry nodes that start perimeter traversal. A special case is that a requesting A-node is riding on the information mesh. Under this circumstance, the requesting A-node performs the perimeter lookup in its every home cell. Note that, since the requesting A-node is already on its home cell perimeter in this case, it can start perimeter traversal directly.



(a) Curly edge (b) Irregular shape

Fig. 7. Cross lookup in an arbitrary sensor network



(a) Curly edge (b) Irregular shape

Fig. 8. Perimeter lookup in an arbitrary sensor network

IV. The Mesh-based Relocation Protocol

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

Throughout the network's lifetime, each A-node maintains a one-hop neighborhood map by beacon messages. Specifically, an A-node locally broadcasts a beacon message carrying its location information on a periodical basis, and meanwhile, it receives beacon messages from its A-node neighbors and liveness reports (see below) from its R-node neighbors. By listening to these periodical messages, the A-node is able to detect new comers, identify failed neighbors, and then update its neighborhood map accordingly. Because this beacon-based neighborhood maintenance mechanism has been employed in many geographic routing protocols such as GFG [1], MSRP may make use of it from the underlying routing protocol rather than implement it.

At initiation, each R-node spontaneously attempts to take a nearest A-node as proxy, by sending that A-node a delegation request. An A-node is allowed to grant a delegation request only when the number of its delegated R-nodes is smaller than a predefined value. R-nodes stay "asleep" most of time during the network's operating period and wake up only at some intervals by a sleep/wakeup protocol. For a R-node having a proxy, it, while being awake, reports its liveness to the proxy node by sending beacon messages and monitors the proxy node's liveness by listening beacon messages. Once a R-node finds that its proxy fails, it moves to replace the proxy node directly. For a R-node without a proxy, it may try to find one during its conscious period. After being chosen as proxy, an A-node executes algorithm DSND to construct an information mesh. 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 (see Definition 5 in Sec. III-A.1) of the failed A-node, by DSND. For brevity, the target proxy of a failed A-node is referred to as *replacement proxy*.

During a replacement discovery process, the two lookup methods, i.e., cross lookup and perimeter lookup, may be selectively used, depending on specified requirement. For the cross lookup method, the northmost, the southmost, the eastmost and the westmost neighbor of a failed A-node, as *server*, send a query message respectively to the north, the south, the east, and the west direction, as shown in Fig. 2. After getting replies, they exchange their discovery results through underlaying routing protocol to find the replacement proxy. For the perimeter lookup method, only the northmost neighbor acts as the A-node's server. 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 *replacement discoverer*.

A replacement discoverer (or discoverer for short) issues a relocation request to the replacement proxy, which then grants the relocation request by sending back an ACK message to the discoverer. After receiving the ACK message, the discoverer starts a shifted node relocation process by sending an action message to the replacement proxy. During this process, the action message is transmitted by protocol GFG [1] to establishes a path, called *relocation path*, from the discoverer to the replacement proxy, and meanwhile, the intermediate nodes along this path start to shift their position toward the failed A-node. More specifically, after sending the action message, the discover moves to the failure node's location, while intermediate nodes moves to the position of its priori hop after forwarding the action message to its next hop. 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. An example of this shifted relocation process is given in Fig. 1. In order not to jeopardize the information mesh or the execution of other network protocols, every relocating node must transfer all the data in its local repository to the new comer at its original position after the relocation process.

V. Implementation Details

A straightforward implementation of the basic protocol design is not sufficient in practice. Some mechanisms must be provided to deal with a number of impact factors such as complex network topology, asynchronous execution, unpredictable node failures, and so on. In this section, we will emphasize on these implementation issues.

A. Maintaining consistency

The information mesh constructed (by DSND) is the key component of protocol MSRP. Its consistency greatly affects the protocol's performance. There are two factors that bring inconsistency to the information mesh. The first one is *late message arrival*, which may be due to asynchronous execution and complex network topology. A common A-node of the registration paths from two proxy nodes can wrongly retransmit the registration message sent by the relatively distant proxy node, because of the late arrival of the one from the close proxy node, violating the blocking rule. Fortunately, this problematic situation can be identified by the common A-node, as soon as it receives both of the two registration messages. The second factor is *proxy resign*. After a proxy node finds that it itself has no more delegated R-nodes available (because of node relocation or node failure), it automatically ceases to be a proxy node. In this case, its information should be removed from the information mesh. This situation is locally been aware of by the proxy node itself.

Inconsistency can be eliminated at the cost of extra control messages. Once an A-node finds the existence of inconsistency in the information mesh, it as initiator starts a *revocation process*, in which the inconsistent proxy node information is erased from the information mesh. More specifically, the initiator sends a revocation message following the forward propagation path of the inconsistent information. The revocation message is processed in exactly the same way as a registration message. It stops at an A-node where the inconsistent information stopped propagating. All the nodes that receive this revocation message remove from their local repositories the information of the proxy node indicated by the message. Such a revocation process can possibly lead to chain effect. That is, the registration messages of the proxy nodes previously blocked due to the revoked information will continue their propagation until the blocking rule is satisfied again at some other A-nodes.

B. Tolerating node failures

Although protocol MSRP is designed to deal with node failures, its execution is not automatically fault-tolerant. One of the impacts from node failures on MSRP is the loss of proxy information during information mesh construction. Similar to the fault-tolerance approach employed by the quorum-based location service [12], [11], MSRP uses thick registration paths to increase information redundancy and thus its fault-tolerance capability. More specifically, during the information mesh construction process, proxy nodes' registration messages are transmitted along paths of certain thickness. For thickness 1, all the A-nodes that overhear a registration message store the embedded proxy location information; for thickness k, these overhearing A-nodes are also required to broadcast the registration message within their $(k-1)$-hop neighborhood.

Another impact from node failures is the loss of control messages for replacement discovery and node relocation. To tolerate such message loss, protocol MSRP uses a simple yet effective fault-tolerance approach, *transmission retrial*. During a replacement discovery process, if a server does not get any reply to its query message, it backs off for a while and retries. In the case that the cross lookup method is used, the server will receive a rely sooner or later, because the network is not partitioned by assumption, and because any failed message forwarding A-node is going to be eventually replaced. However, In the case that the perimeter lookup method is applied, this transmission retrial mechanism may cause waiting loops. To avoid dead-lock, during perimeter traversal, a A-node is required to send the query message back to the entry point right away if it finds that the next hop has failed, such that the requesting A-node does not need to wait. During a node relocation process, if the replacement discoverer does not receive a reply (an ACK message) from the replacement proxy to its relocation request, it reissues the request. If, after a predefined number of trials, it still does not get any reply, it considers that the replacement node is unavailable and then tries to discover another one.

Run-time node failures may possibly ineffect the objective of MSRP. Specifically, if all the default servers (i.e., the directional foremost A-node neighbors) of a failed A-node fail, or if the replacement discoverer fails, the failed A-node can not be replaced according to the protocol design. Therefore, MSRP requires that, every other A-node neighbor of the failed A-node monitor the default servers via underlying routing protocol until the failed A-node is actually replaced. If a server fails during the monitoring period, the second foremost A-node neighbor in the corresponding direction takes over immediately. Note that, if some intermediate A-nodes along a relocation path fail in a node relocation process executed for a failed A-node, the failed A-node is replaced by the replacement discoverer due to the nature of the shifted node relocation method anyway, while those failed intermediate A-nodes will later be replaced with some other R-nodes by protocol MSRP.

C. Discovering relocation path

Relocation path discovery is in essence a QoS routing process started by a replacement discoverer. Its objective is to establish a path, between a replacement discoverer and a replacement proxy, that 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 shit their position toward a failed A-node. From energy-saving point of view, node relocation should involve shortest total moving distance and 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, protocol MSRP takes the concept of COST over PROGRESS ratio [13] as routing criterion and copes it with routing protocol GFG [1] to discover relocation path. In MSRP, the COST is defined as the length of the considered next hop, and the 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 node, i.e., a replacement discoverer, and by a_i the i-th hop along the path from a_0 to destination node 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:

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

D. Solving relocation contention

Because of the distributed nature of protocol 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 failure 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. Recall that, a node relocation process is triggered by a replacement discoverer after its relocation request is granted by the replacement proxy, i.e., the proxy of the replacement. MSRP requires that, once a proxy node grants some A-node's relocation request, it reject all the upcoming requests, preventing type-I node contention from happening. As for an replacement discoverer whose relocation request is rejected, it backs off for a while and then tries to find some other R-node by a new replacement discovery process. An A-node in a relocation path will start to relocate as soon as it forwards the corresponding action message. If the A-node also belongs to another relocation path, then the decision on where it should move depends on which path it receives an action message first along. After moving, the A-node will transfer all the local data to the new comer at its original position, which then respond to buffered or upcoming action messages. By this means, type-II node contention is solved properly.

VI. Protocol Analysis

In this section, we are going to analyze the characteristics of our new protocol MSRP, and show its advantages in comparison with the three existing relocation protocols, i.e., WCP [14], WCPZ [16] and ZONER [10].

Both MSRP and ZONER are a localized algorithm because they require sensors to know merely about their own neighborhood information and do not involve any global computation like networkwide flooding or clustering, while both WCP and WCPZ involve certain centralized control (for mobile node management or for cluster management) and thus belong to the quasi-distributed algorithm category. Further, WCPZ relies on the preknowledge of the sensor field for grid formation and relocation path discovery. Considering scalability and applicability, the localized and preknowledge-independent protocols MSRP and ZONER are more desirable than the other two protocols WCP and WCPZ, especially for dense wireless sensor networks deployed in unknown environments.

Because the goal of a sensor relocation protocol is to maintain coverage, guaranteed node replacing is a crucial evaluation criterion. Both MSRP and ZONER employ the face routing technique to pass messages around void areas appearing in the network topology. The void-area tolerance ability ensures replacement discovery and relocation path discovery, and therefore successful node replacing. On the contrary, because both WCP and WCPZ do not uses the face routing technique, they become problematic in the face of void areas. Considering the guaranteed node replacing property, MSRP and ZONER defeat the other two protocols already.

A sensor relocation protocol is expected to have constant per node storage load $O(1)$, considering the sever resource constraints of wireless sensor networks. MSRP does possess this property by the protocol description in Sec. IV and Theorem 3, while the three existing protocols do not. In WCP, if all the mobile nodes are compactly located in a small area, some static nodes may be within the advertisement (flooding) range of every mobile node and thus have to store the information of all the mobile nodes. Similarly, in WCPZ and ZONER, if all the R-nodes are horizontally or vertically collinear, the A-nodes along a quorum have to store the information of all the R-nodes. Thus, WCP, WCPZ and ZONER all require memory space $O(n')$ on each sensor node. The big different in storage space requirement shows the unbeatable advantage of MSRP over the other relocation algorithms.

In all the four protocols, majority of the messages are generated for replacement discovery. In a network with arbitrarily bad topology, bot MSRP and ZONER can work probably with increased message overhead (due to the application of the face routing technique), but the effectiveness of protocols WCP and WCPZ can not be guaranteed. Under this circumstance, to have a clear and fair comparison on message overhead, we only consider a network with square-grid-like topology, where greedy forwarding always works. Hence, the message complexity of MSRP for replacement discovery can be expected to be much less than $O(\nu\sqrt{n})$ where $\nu \leq Min\{n', n\}$ is the number of proxy nodes according to Theorem 2, and those of protocols WCPZ and ZONER are expected to be $O(n'\sqrt{n})$ due to their quorum-based replacement discovery method. As for protocol WCP, it uses a simple restricted flooding-based node discovery method, which is questionable because of the difficulty in the predefinition of the size of flooding areas. If the size of each flooding area is proportional to the size of the network, the message complexity of this method can be expected to be as bad as $O(n'n)$. By above analysis, we can see that, from message complexity, point of view, MSRP performs at least as well as the three existing protocols in worst case but obviously better than them in average case.

Relocation delay and energy consumption are important evaluation metrics. Protocol WCP moves mobile nodes directly to sensing holes, while protocols MSRP, WCPZ and ZONER all relocates a replacement node to the location of a failed A-node in a shifted manner. By the direct relocation method, the relocation distance can be as bad as the spatial diameter (bounded by $O(n)$, where n is a non-constant value) of the network. Therefore, WCP can cause the battery power of a mobile node over-consumed and generate non-constant relocation delay. By the shifted relocation method, all the the nodes along a relocation path shift their position toward a failed A-node, as shown in Fig. 1. In this process, a node's moving distance is always bounded by its communication radius cR, a constant value. Hence, MSRP, WCPZ and ZONER have balanced energy consumption and constant relocation delay (in the case of simultaneous shifting). Mentionably, although these three protocols use a similar shifted relocation method, the difference between the ways that they discover a relocation path make them actually perform differently. MSRP uses an advantageous localized routing mechanism to establish a relocation path between a replacement and a failed A-node, while WCPZ employs a undesired flooding-based routing mechanism to do so. As for ZONER, it integrates relocation path discovery within replacement discovery processes for message-saving purpose but without energy-saving consideration.

A replacement selected by ZONER for a failed A-node is always a R-node geographically closest to the failed A-node, while this may not be the case for WCP, WCPZ and MSRP. In WCP, void areas caused, for example, by physical obstacles can block the advertisement of mobile sensors, causing the

	MSRP	**WCP [14]**	**WCPZ [16]**	**ZONER [10]**
Protocol Nature	localized	quasi-distributed	quasi-distributed	localized
ZERO Preknowledge Requirement	yes	yes	no	yes
Guaranteed Node Replacing	yes	no	no	yes
Constant Relocation Latency	yes	no ($O(n)$)	yes	yes
Constant Per-node Storage Load	yes	no ($O(n')$)	no ($O(n')$)	no ($O(n')$)
Replacement Discovery Method	mesh	flooding	quorum	flooding & quorum
Guaranteed Nearby Replacement Discovery	yes	no	no	yes
Closest Replacement	no	no	no	yes
Message Complexity	$\leq O(\nu\sqrt{n})$	$O(n'n)$	$O(n'\sqrt{n})$	$O(n'\sqrt{n})$
Node Relocation Method	shifted	direct	shifted	shifted
Guaranteed Relocation Path Discovery	yes	—	no	yes
Energy-aware Relocation Path	yes	—	yes	no

$\nu \leq Min\{n', n\}$.

TABLE I
PROTOCOL COMPARISON

loss of the closest replacement property. For the same reason, WCPZ may fail to find the nearest R-node as replacement for a failed A-node. In MSRP, the Euclidean distance from a replacement node to a failed A-node can only be guaranteed to be bounded by twice the Euclidean distance between the failed A-node and its closest R-node, according to Theorem 1. For WCP, the lack of the closest replacement property is indeed a major drawback because the direct relocation method that it uses is vulnerable to Euclidean distance. However, for WCPZ and MSRP, we do not consider so, because the shifted node relocation method employed by them weakens the gravity of Euclidean distance. In particular, it is a tradeoff with all the other nice properties for MSRP.

Table I comparatively lists the characteristics of the four relocation protocols MSRP, WCP, WCPZ and ZONER. Note that, to have fair comparison, message complexity is only for a network with square-grid-like topology, where greedy forwarding always works. From this table, we can clearly see that our new protocol MSRP achieves obvious strength in many aspects by loosing the requirement on the "closest replacement" property, and that, to date, no existing relocation protocol is comparable with it. In this case, simulation-based performance evaluation is not necessary.

VII. CONCLUSION

In this paper, we first presented a Distance Sensitive Node Discovery algorithm (DSND) based on a novel structure, *information mesh*. Then on the basis of DSND, we proposed a localized Mesh-based Sensor Relocation Protocol (MSRP) for mobile sensor networks. MSRP does not require any pre-knowledge of the sensor field. Its objective is to replace failed sensors with the redundant ones scattered in their vicinity through autonomous and strategic nodal movement. MSRP can be used as fault-tolerance approach to maintain constant sensing coverage in the presence of unpredictable node failures. It is superior to the existing relocation protocols[14], [16], [10] in many aspects, as shown in Tab. I. We notice that, if the nodes within the communication range of a failed A-node fail all together, MSRP can not replace the failed A-node. However, this is a non-trivial problem for all the existing relocation schemes. We take it as our future work.

In addition, the sub-algorithm DSND can be easily extended to solve the distance-sensitive service discovery problem, that is, discover, within the vicinity of a service consumer, a service provider that can deliver the requested service with reasonable delay. An instance of the distance-sensitive service discovery problem is that, in a wireless and actor network, when an event occurs, a closest or satisfactorily closest actor must be relocated to the event location. A possible extension to DSND algorithm for this problem is the following: service providers construct a multi-layered information mesh; different layers correspond to different types of services; when a consumer wants to discover a particular type of service, it simply executes DSND algorithm in the corresponding layer of the information mesh. Formalizing DSND as a service discovery algorithm is also part of our future work.

REFERENCES

[1] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. "Routing with Guaranteed Delivery in Ad Hoc Wireless Networks". In *Proc. of ACM DIALM*, pp. 48-55, 1999.

[2] H. Frey and I. Stojmenovic. "On Delivery Guarantees of Face and Combined Greedy-Face Routing Algorithms in Ad Hoc and Sensor Networks". In *Proc. of ACM MobiCom*, 2006.

[3] A. Gallais, J. Carle, D. Simplot-Ryl and I. Stojmenovic. "Localized Sensor Area Coverage with Low Communication Overhead". In Proc. PerCom, 2006.

[4] N. Heo and P. K. Varshney. "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] A. Howard, M. J. Mataric, and G. S. Sukhatme. "Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem". In *Proc. of DARS*, pp. 299-308, 2002.

[6] J. W. Jaromczyk and G. T. Toussaint. "Relative neighborhood graphs and their relatives". In *Proc. of the IEEE*, vol. 80, pp. 1502-1517, 1992.

[7] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. "Geometric ad-hoc routing: Of theory and practice". In *Proc. of ACM PODC*, pp. 63-72, 2003.

[8] B. Leong, S. Mitra, and B. Liskov. "Path vector face routing: Geographic routing with local face information". In *Proc. of IEEE ICNP*, 2005.

[9] X. Li and N. Santoro. "An Integrated Self-Deployment and Coverage Maintenance Scheme for Mobile Sensor Networks", In *Proc. of MSN*, LNCS 4325, pp. 847-860, 2006.

[10] X. Li and N. Santoto. "ZONER: A ZONE-based Sensor Relocation Protocol for Mobile Sensor Networks". In *Proc. of IEEE LCN/WLN*, pp. 923-930, 2006.

[11] D. Liu, I. Stojmenovic and X. Jia. "A scalable quorum based location service in ad hoc and sensor networks". In *Proc. of IEEE MASS*, 2006.

[12] I. Stojmenovic. "A scalable quorum based location update scheme for routing in ad hoc wireless networks". Technical Report, SITE, Univ. of Ottawa, TR-99-09, 1999.

[13] I. Stojmenovic. "Localized network layer protocols in wireless sensor networks based on optimizing cost over progress ratio". IEEE Network, 20(1):21-27, 2006.

[14] G. Wang, G. Cao, and T. L. Porta. "Proxy-Based Sensor Deployment for Mobile Sensor Networks". In *Proc. of IEEE MASS*, pp. 493-502, 2004.

[15] G. Wang, G. Cao, and T. L. Porta. "Movement-Assisted Sensor Deployment". In *Proc. of IEEE INFOCOM*, vol. 4, pp. 2469-2479, 2004.

[16] G. Wang, G. Cao, T. L. Porta, and W. Zhang. "Sensor Relocation in Mobile Sensor Networks". In *Proc. of IEEE INFOCOM*, pp. 2302-2312, 2005.

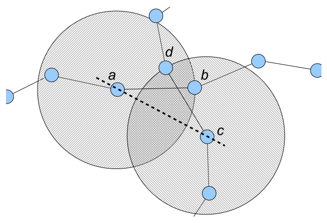
[17] K. Wu, Y. Gao, F. Li, and Y. Xiao. "Lightweight Deployment-Aware Scheduling for Wireless Sensor Networks". In ACM MONET Journal, 2005.

[18] J. Wu and S. Yang. "SMART: A Scan-Based Movement-Assisted Sensor Deployment Method in Wireless Sensor Networks". In *Proc. of IEEE INFOCOM*, vol. 4, pp. 2313- 2324, 2005.

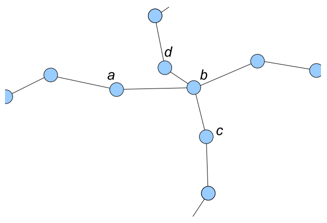
[19] G. Xing, C. Lu, R. Pless, and Q. Huang. "Impact of sensing coverage on greedy geographic routing algorithms". IEEE Tran. on Parallel and Distributed Systems, 17(4):348-360, 2006.

[20] Y. Zou and K. Chakrabarty. "Sensor deployment and target localization in distributed sensor networks". ACM Tran. on Embedded Computing Systems, 3(1):61-91, 2004.

APPENDIX
DESCRIPTION OF INTERSECTION TRANSFORMATION



(a) Before transformation



(b) After transformation

Fig. 9. An example of intersection transformation

A link-crossing situation can be locally transformed to a node-sharing situation in a sensor network modeled as a unit disk graph. Consider two crossing links $\overline{ab}$ and $\overline{cd}$, as shown in Fig. 9(a). Randomly take one node from each link, say a from $\overline{ab}$ and c from $\overline{cd}$. If a and c are neighboring each other, then we are done simply by replacing $\overline{ab}$ with two links $\overline{ac}$ and $\overline{cb}$. If a and c are not each other's neighbor, then the other two nodes b and d must reside on the same side of the line ac and within the intersection area of the communication ranges of nodes a and c in order for $\overline{ab}$ and $\overline{cd}$ to intersect across. This can be easily figured out by examining the example given in Fig. 9(b). In this case, we can replace $\overline{cd}$ ($\overline{ab}$) with two links $\overline{cb}$ and $\overline{bd}$ (resp., $\overline{ad}$ and $\overline{db}$), finishing the transformation.