VAPR: Void-Aware Pressure Routing for Underwater Sensor Networks

Youngtae Noh, *Member*, *IEEE*, Uichin Lee, *Member*, *IEEE*, Paul Wang, *Member*, *IEEE*, Brian Sung Chul Choi, *Member*, *IEEE*, and Mario Gerla, *Fellow*, *IEEE*

Abstract—Underwater mobile sensor networks have recently been proposed as a way to explore and observe the ocean, providing 4D (space and time) monitoring of underwater environments. We consider a specialized geographic routing problem called pressure routing that directs a packet to any sonobuoy on the surface based on depth information available from on-board pressure gauges. The main challenge of pressure routing in sparse underwater networks has been the efficient handling of 3D voids. In this respect, it was recently proven that the greedy stateless perimeter routing method, very popular in 2D networks, cannot be extended to void recovery in 3D networks. Available heuristics for 3D void recovery require expensive flooding. In this paper, we propose a Void-Aware Pressure Routing (VAPR) protocol that uses sequence number, hop count and depth information embedded in periodic beacons to set up next-hop direction and to build a directional trail to the closest sonobuoy. Using this trail, *opportunistic directional forwarding* can be efficiently performed even in the presence of voids. The contribution of this paper is twofold: 1) a robust soft-state routing protocol that supports opportunistic directional forwarding; and 2) a new framework to attain loop freedom in static and mobile underwater networks to guarantee packet delivery. Extensive simulation results show that VAPR outperforms existing solutions.

Index Terms—Pressure routing, anycast, opportunistic routing

1 Introduction

Underwater acoustic sensor networks have lately been suggested as a potent means of supporting aquatic applications ranging from environmental monitoring to intrusion detection [1], [2], [3]. A large number of mobile sensor nodes are deployed in the region of interest to form an *ad-hoc* network (called SEA Swarm) for short-term acoustic exploration. For instance, mobile sensors can track the dispersion in space and time of oil spill plumes escaping from a broken oil pipe (e.g., Deepwater Horizon oil spill). In a SEA Swarm, each node is equipped with a variety of sensors and a low-bandwidth acoustic modem. Moreover, each node has a fish-bladder like apparatus and a pressure gauge, and its depth can be configured when deployed (e.g., Drogues [4]). A swarm of sensor nodes is escorted by sonobuoys on the sea surface, where sonobuoys are equipped with both acoustic and radio modems (Wi-Fi or satellite communications) and GPS. Each sensor node in the swarm reports relevant data to any one of the sonobuoys with acoustic multihop routing (called *anycast routing*); the data can then be offloaded to a monitoring center via radio communications for further offline processing. In a GPS-denied underwater environment, the need for global, distributed localization for sensor data geotagging is relaxed via offline, approximate localization at a monitoring center that uses local distance measurements or distance estimates from sonobuoys (collected along with sensor data) [5], [6].

Our goal in this paper is to design an efficient anycast routing protocol for underwater data collection that addresses several challenges unique to underwater communications. Most notably, the underwater acoustic channel is severely constrained by long propagation latency and low bandwidth (usually less than 100 Kbps) [7], and is prone to packet losses and collisions in a congested network. Energy efficiency is a critical factor as well, given that acoustic transmissions consume far more energy than terrestrial radio communications (reception to transmission power ratio of 1:125 [8]).

One may modify existing terrestrial routing protocols in mobile underwater networks (e.g., OLSR [9], DSDV [10], AODV [11], DSR [12]) to support anycast routing by assigning a single virtual node ID to all sonobuoys [13]. However, the major shortcomings of this approach are twofold at least: 1) these protocols require frequent systematic flooding and route maintenance with neighboring nodes, which are very expensive operations under water, and 2) it is challenging to incorporate opportunistic forwarding mechanisms (e.g., ExOR [14], LCOR [15]) into the stateful routing protocols due to node mobility [16]—under unreliable acoustic channels, opportunistic forwarding can combat packet losses by taking advantage of simultaneous packet reception among one node's neighbors.

- *Y. Noh is with the Department of Computer Science, University of California, Los Angeles, 3803B Boelter Hall, Los Angeles, CA 90095. E-mail: ytnoh@cs.ucla.edu.*
- *U. Lee is with the Department of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology (KAIST), 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Korea. E-mail: uclee@kaist.edu.*
- *P. Wang is with Jet Propulsion Laboratory, M/S 301-480, 4800 Oak Grove Drive, Pasadena, CA 91109. E-mail: Paul.Wang@jpl.nasa.gov.*
- *B.S.C. Choi is with Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043. E-mail: sungchoi@google.com.*
- *M. Gerla is with the Department of Computer Science, University of California, Los Angeles, 420 Westwood Plaza, Los Angeles, CA 90095. E-mail: gerla@cs.ucla.edu.*

Manuscript received 3 Aug. 2011; revised 5 Jan. 2012; accepted 14 Feb. 2012; published online 21 Feb. 2012.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2011-08-0437.
Digital Object Identifier no. 10.1109/TMC.2012.53.

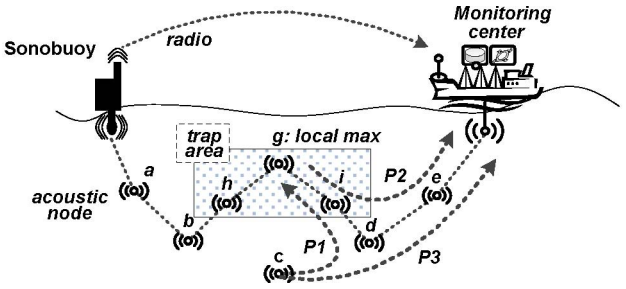


Fig. 1. Conventional pressure routing in SEA Swarm.

Therefore, recent research in underwater networks has been directed to position-based or geographic routing because it does not require any link state exchanges or route maintenances (i.e., *stateless* and *localized*). Another key advantage of geographic routing is that it enables localized geo-opportunistic forwarding; i.e., a subset of nodes that have correctly received the packet can collaboratively schedule a packet transmission to *maximize* its progress toward the destination [17], [16]. Note that our geographic routing problem is specialized in that it is anycast to any buoy on the surface. Thus, it suffices to route a packet upwards to shallower depths. Given that the onboard hydraulic pressure gauge can accurately estimate depth (avg. error < 1 m [18]), we can use depth information for geographic anycast routing (called *pressure routing*) [19], [16].

Despite its benefits, simple greedy pressure routing often fails in *sparse* underwater networks due to the presence of 3D *voids*—packets must be routed around such routing holes. As depicted in Fig. 1, a data packet originating from node c may eventually be routed to a local maximum node g when greedily forwarded based on depth (via path P1). Node g cannot make any progress toward the surface because it does not have any neighboring node with depth shallower than its own. Node g must thus perform a route recovery process to get around the void via path P2. For 3D networks, however, it has been proven that there is no efficient memoryless routing algorithm that delivers messages deterministically in 2D face routing [20], which is also true for pressure routing. Researchers, therefore, have proposed several heuristic recovery methods such as random walks [21] and 2D void surface flooding [16].

There are at least two major drawbacks of such heuristic recovery methods: 1) the fallback mechanism must discover and maintain recovery paths, which are expensive in mobile networks, even more so in an underwater environment; and 2) some of the nodes around a void area will eventually route packets to local maxima (called *trapped nodes*, e.g., node i and h in Fig. 1), and any nodes located beneath the trapped nodes can potentially suffer from route hop stretch because local greedy forwarding may lead packets to local maxima and then invoke fallback mechanisms. In Fig. 1, node c's packet is greedily forwarded to the local maxima via path $P1$ and then is rerouted via path $P2$ (total 7 hops), whereas this packet can be directly delivered via path $P3$ (total 3 hops). Note that these problems will be more pronounced when the number of sonobuoys is sparse or when node density is low (both cases incur more voids in the network).

This serious shortcoming of pressure routing is inherently due to the nodes' blindness to the network topology, as they make localized routing decisions. In terrestrial stateful routing (e.g., DSDV, OLSR), each node has a *global view* of the network topology, with which packets can always be efficiently routed via shortest paths, at the cost of energy-hungry route discovery and maintenance mechanisms. This observation suggests that there is a tradeoff between routing efficiency and route maintenance cost. In this paper, we improve pressure routing by providing nodes with a partial view of the network topology such that greedy pressure routing is guided by *soft-state breadcrumbs* (i.e., *up/down* directions) from the sonobuoys; this method completely obviates the need of handling voids with heuristic methods.

This soft-state breadcrumb approach, which exploits periodic beaconing to build directional trails toward the surface, is much more efficient and robust than conventional underwater pressure routing protocols in that nodes maintain an enhanced view of network topology without incurring the extra cost of energy-hungry route discovery and maintenance mechanisms; nodes utilize *geo-opportunistic forwarding* along the directional trails. In this paper, we make the following contributions:

- We propose the Void-Aware Pressure Routing (VAPR) protocol that uses surface reachability information to set up each node's next-hop direction toward the surface through which local *opportunistic directional forwarding* can always be used for data packet delivery even in the presence of voids. VAPR takes advantage of geo-opportunistic forwarding and is very robust to network dynamics such as node mobility and failure. VAPR neither requires additional recovery path maintenance nor incurs any hop stretch caused by the recovery fallbacks in existing solutions [21], [16].
- We provide a new framework of attaining loop freedom using our soft-state breadcrumb approach in mobile networks. Also, we perform extensive simulations and verify that VAPR's enhanced beacon based directional forwarding outperforms existing pressure routing protocols (e.g., DBR [19] and HydroCast [16]) and a simple hop-based greedy routing protocol under the scenarios considered.

This paper significantly enhances our preliminary work [22] in that we include

1. a thorough review of underwater pressure routing protocols and route recovery techniques (Section 2),
2. an enhanced protocol design and elaborate description of the proposed protocol (Section 4),
3. a detailed discussion on the loop-free property (Section 4.3), and
4. extensive simulation results of the proposed protocol, incorporating the Meandering Current Mobility (MCM) model under various system parameter configurations (Section 5.2).

The rest of this paper is organized as follows: in Section 2, we review the related work in the field. In Section 3, we provide a brief overview of VAPR. In Section 4, we provide

a detailed description of VAPR. In Section 5, we validate the performance of VAPR by comparing it with existing approaches. In Section 7, we conclude the paper and discuss future work.

2 Related Work

Underwater routing protocols. Pompili et al. [23] proposed two routing protocols for delay-sensitive and delay-insensitive applications in a 3D underwater environment. The delay-sensitive routing protocol is based on virtual circuit routing. Primary and backup multihop node-disjoint data paths are calculated by a centralized controller to achieve an optimal delay. The delay-insensitive routing protocol is a distributed geographic solution aimed at minimizing the energy consumption via back-to-back packet transmissions and cumulative acknowledgments. Vector-Based Forwarding (VBF) [24], [25] prescribes that packets be forwarded to the nodes that are located within a route of the given width between the source and the destination. This relay selection algorithm saves energy consumption by reducing the number of packet relays. Note that there are also geographic routing protocols that exploit the opportunistic forwarding features in underwater environments [26], [19], [16], which will be detailed later. Vieira et al. [27] proposed Phero-Trail routing that efficiently delivers packets to a mobile sink by following a pheromone trail of the sink. Besides unicast routing and converge-cast routing, broadcasting is also required by some underwater sensor applications (e.g., reprogramming sensor nodes). Casari and Harris [28] proposed several reliable broadcasting protocols that leverage the ability to use small bands to transmit an alert packet for a long distance. After sending alert signals, nodes reduce the Transmission Range (TR) and select only certain neighboring nodes in order to repeat broadcast, thereby lowering the total number of transmissions required. Similar ideas can be found in other related work [29], [30]. Readers can find more detailed survey of recent underwater routing protocols in the survey papers [31], [32].

Opportunistic routing. Most opportunistic routing protocols (also called anypath routing) such as ExOR [14], Least Cost Opportunistic Routing (LCOR) [15], which do not use geographic information, require global topology and link quality information (like link state routing) to find a set of forwarding groups toward the destination; thus, they are more suitable for *static* wireless mesh or sensor networks. In practice, geographic routing can also benefit from opportunistic forwarding, as in Geographic Random Forwarding (GeRaF) [17], Contention-Based Forwarding (CBF) [33], and Focused Beam Routing (FBR) [26], though these methods are not optimal due to lack of global knowledge. In the literature, researchers have typically used a geometric shape (e.g., a triangle/cone [33], [26]) that is *faced toward the destination* for forwarding set selection to prevent hidden terminal problems.

Pressure routing. Yan et al. proposed a greedy anycast routing solution called Depth-Based Routing (DBR) [19]. They suggested that packet forwarding decisions be made locally based on the pressure (or depth) level measured at each node. Packets would then be geographically forwarded to nodes with shallower depth in a greedy fashion toward the water surface. This hydraulic pressure-based anycast routing protocol benefits from being stateless and does not require expensive distributed localization [34], [35]. DBR exploits the simultaneous packet reception induced by the broadcast nature of the wireless medium and performs *opportunistic greedy forwarding* via a subset of the neighbors that have received the packet correctly. However, DBR lacks an efficient forwarding set selection method and a recovery method from local maxima. Lee et al. proposed HydroCast [16] to remedy these problems. HydroCast improves the efficiency of the forwarding set selection method by choosing a set that maximizes greedy progress yet limits cochannel interference. Additionally, HydroCast has a route recovery scheme that uses a hop limited ring search over the 2D surface of a convex hull around the void area to discover a recovery path. Like these protocols, our protocol relies on opportunistic greedy forwarding with a forwarding set selection algorithm borrowed from HydroCast.

Recovery mechanisms. The techniques for routing around the local maxima can be classified into two categories: *stateless* (memoryless) and *stateful*. Recently, Durocher et al. [20] proved that stateless recovery in 3D networks is infeasible unless it is as naïve as random walks [21]. There was an attempt to project a 3D network onto a 2D plane [36], but it was shown that face routing on the projected 2D plane cannot guarantee packet delivery. Liu et al. proposed a Partial Unit Delaunay Triangulation (PUDT) algorithm to construct hulls that partition the 3D network into subspaces so that recovery can be simply done by exploring the subspace. Zeng et al. [37] proposed to embed the 3D network into a hyperbolic space using a discrete hyperbolic Ricci flow. Nodes are mapped to virtual coordinates in the hyperbolic space, which intrinsically have paths to avoid holes—greedy forwarding is always possible in hyperbolic space. However, in mobile underwater sensor networks, this mapping must be periodically refreshed (due to mobility), thus leading to de facto flooding. Thus, the cost is comparable to that of flooding heuristics.

Several *stateful* approaches have been suggested mainly for 2D networks, but they are generally extensible also to 3D networks [38], [39], [40], [41], [37]. He et al. proposed SPEED, which reactively uses backpressure-based backtracking to inform upstream nodes to prune paths that reach a local maximum [38]. In [39] and [42], a spanning tree was used in which each node has an associated convex hull that contains within it the locations of all its descendant nodes in the tree. Liu and Abu-Ghazaleh [40] proposed using a virtual coordinate system to route packets, in which a packet can be backtracked toward one of the anchor nodes in the event that recovery is needed. Geo-LANMAR [41] inherits the group motion support of Landmark Routing (LANMAR) to identify landmark nodes (cluster-heads) and maintains routes to such landmarks using a combination of georouting and directional forwarding. A periodic beacon propagates, from which the geodistance, hop distance and cluster membership are derived by each landmark, thus functioning as a kind of a distributed DNS. Each node extracts from the beacon the direction to the landmark along the beacon traced shortest path. When georouting gets stuck, the beacon-guided direction is used (as in directional forwarding) to recover from voids. The direction to a landmark is used because next-hops change too rapidly in a mobile environment, whereas the direction changes occur less frequently.

Key differences. VAPR is also stateful and resembles Geo-LANMAR in that sonobuoys propagate surface reachability information (via enhanced beaconing) for each underwater node to setup its next-hop direction toward the surface. The key difference between VAPR and previous schemes are threefold: 1) VAPR always uses *local greedy directional forwarding* for data delivery on the basis of the direction cues (it does not wait until it gets stuck in a local maximum like Geo-LANMAR does); 2) VAPR neither requires recovery path maintenance nor incurs any hop stretch caused by the recovery fallbacks when compared to existing solutions [21], [16] and; 3) VAPR requires a small soft-state, i.e., next-hop direction and hop distance information at each node (as readily available from the Beacon) and is robust to network dynamics such as node mobility, failures and possible sleep/wake up cycles.

3 VAPR Overview

VAPR is composed of two major components, namely *enhanced beaconing* and *opportunistic directional data forwarding*. In the former, sonobuoys propagate their reachability information to sensor nodes via enhanced periodic beaconing.[1] In enhanced beaconing, each node's beacon is augmented with additional information, namely the sender's depth, hop count to a sonobuoy, sequence number, and its current *data forwarding direction* (toward the surface). When receiving the augmented beacons from predecessors, each node updates its variables, namely *minimal hop to the surface*, *sequence number*, *data forwarding direction*, and *next-hop data forwarding direction* (i.e., predecessor's data forwarding direction).

In the beginning, every sonobuoy on the surface initializes these variables and starts beaconing. After receiving a beacon message, a node can tell whether it has received the message from deeper or shallower depth, and each node sets its data forwarding direction toward the surface. The direction is set as *up* when a beacon is received from a shallower depth node; otherwise, it is set as *down*. When multiple direction cues from different sonobuoys are received, the direction cue with the minimal hop count is chosen. Also, a node's next-hop data forwarding direction is set based on the beacon sender's data forwarding direction. For instance, in Fig. 1, since node d/e receives a beacon message from a shallower depth node, it sets its data forwarding direction and next-hop data forwarding direction as *up-up*, respectively. In contrast, when node i receives a beacon message from a deeper depth node (i.e., node d), it sets its data forwarding direction and next-hop data forwarding direction as *down-up*, respectively. After updating its local states, each node prepares a beacon message to broadcast by incrementing the hop count and setting its current depth, data forwarding direction, and sequence number. This beaconing process will repeat, and thus, nodes essentially build directional trails toward their closest sonobuoys on the surface. Note that a direction change is caused by *voids*; e.g., in Fig. 1, path $P3$ has only a single direction (*up*), whereas path $P2$ experiences a direction change due to the void (*down* and *up*).

1. Recall that periodic beaconing is an essential part of the architecture for underwater localization. It comes at zero incremental cost, and it is up to the specific routing scheme to exploit it or not.

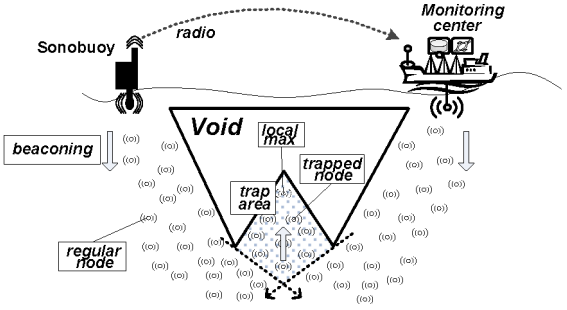


Fig. 2. Terminology.

Given this, VAPR performs local *opportunistic directional forwarding* to deliver data according to the directional trails. The forwarding decision is solely made based on the local state variables, namely the data forwarding direction and next-hop data forwarding direction—hop-count information is never used for routing to exploit opportunistic packet receptions. Two cases are possible: 1) if there is no void, packets can always be greedily routed via the upward direction, and we can solely rely on the data forwarding directions for routing; and 2) if there are voids, there will be direction changes, and the next-hop forwarding direction is jointly used with the data forwarding direction to guide the routing direction. Consider the $\bigvee$-shape topology in Fig. 1 and assume that the states of nodes g, i, d, and e are *down-down*, *down-up*, *up-down*, and *up-up*, respectively. A data packet that originated from node g can be greedily routed downwards to node i. This node realizes that there is a direction change in the next hop (as its status is *down-up*), and for packet forwarding it only considers the neighboring nodes whose depth levels are deeper and whose data forwarding directions are upward (i.e., node d), ensuring that a change of the routing direction is correctly made.

The following terminology is used throughout the paper (see Fig. 2). A *local maximum* node is a node whose depth level is shallower than that of all its neighboring nodes but deeper than that of the sonobuoys; local Greedy Upward Forwarding (GUF) cannot make any progress toward the surface. A *trapped node* is a node in which greedy forwarding eventually leads to a local maximum node. A local maximum node is also by definition a trapped node. As shown in the figure, trapped nodes are usually found beneath the concave area of voids. The area in which trapped nodes reside is called the *trap area*. The rest of the nodes (that are not trapped nodes) are called *regular nodes*.

4 VAPR Details

4.1 Enhanced Beaconing

In VAPR, each sonobuoy propagates surface reachability information to underwater nodes to give nodes an enhanced view of the network. We modify periodic beaconing of pressure routing (originally designed to exchange hello messages amongst neighbors) by embedding the sender's depth, hop count, data forwarding direction, and sequence number in a beacon message. Given this information, each node i keeps its local status of $node_{ID}(i)$, $\pi(i)$, $DF\_dir(i)$, $NDF\_dir(i)$, and a tuple of $hop\_count(i)$ and seq_num, where $node_{ID}(i)$ is node i's ID,

TABLE 1
Terminology

Terms	Definitions
V	Set of nodes
S	Set of sonobuoys, where $S \subset V$
$\pi(node)$	Water pressure measured at $node$
DF_dir	$\in \{null, down, up\}$
NDF_dir	$\in \{null, down, up\}$
hop_count	$\in [0, \infty]$
seq_num	Sequence number used for periodic beaconing

$\pi(i)$ is i's depth from the sea surface (or pressure level), $DF\_dir(i)$ is i's data forwarding direction toward a sonobuoy, $NDF\_dir(i)$ is the next-hop data forwarding direction (i.e., data forwarding direction of i's predecessor), $hop\_count(i)$ is the number of hops from i to the sonobuoy, and seq_num is the sequence number used for periodic beaconing. Here, we assume that sonobuoys on the surface are equipped with GPS, through which their clocks are synchronized, and that they use the same sequence number for periodic beaconing. The sequence number will be incremented periodically; e.g., with a fixed Beacon Interval (BI) of 30 s. As we will see, the sequence number and hop count allow nodes to handle potential direction loops or oscillations caused by nodal mobility and randomness of periodic beaconing, which will be further discussed in Section 4.3. Also, each node maintains minimal information about its one-hop neighbors; i.e., for each neighbor n, we keep $node_{ID}(n)$, $\pi(n)$, $DF\_dir(n)$. Every neighbor entry will be refreshed whenever a beacon message is received from that node. If the timer of an entry expires, the expired entry can be deleted from the current node's storage, thereby removing the node from its neighboring node set.

Algorithm 1 is used to initialize each nodes' internal states (see Table 1 for the terminology used in the pseudocodes in this paper). Initially, each node in the network begins as an isolated local maximum node (i.e., indicated by $hop\_count(node)$ equaling ∞, which we explain later), with the exception of sonobuoys on the sea surface (as they are the destinations). Naturally, each node in the connected component with at least one sonobuoy will have its status changed to a nonlocal maximum node.

Algorithm 1. Routing initialization

```
1: procedure Initialize(node)
2: if node ∈ S then
3:     DF_dir(node) ← up
4:     hop_count(node) ← 0
5:     seq_num(node) ← 0
6: else
7:     π(node) ← pressure measured at node
8:     DF_dir(node) ← null
9:     NDF_dir(node) ← null
10:    hop_count(node) ← ∞
11:    seq_num(node) ← null
12: end if
13: end procedure
```

Algorithm 2 is used to broadcast periodic beacons and handle received beacons. In a beacon message, nodes embed their local states, namely current hop-count, sequence number, depth, and data forwarding direction. To minimize the chance of collisions and synchronization, nodes add random jitters for periodic beaconing using timers when they broadcast beacon messages; then, the nodes set up a new timeout for the next beaconing. After receiving a beacon message, each node checks the validation of the received beacon by checking the sequence number (increasing) and hop-count (smaller), and each node sets its data forwarding direction and updates its next-hop data forwarding direction. As illustrated earlier, the direction is set as *up* if a beacon message is received from a shallower depth node; otherwise, it is set as *down*. After the data forwarding direction is set, a node's next-hop data forwarding direction is also updated based on the data forwarding direction of the beacon sender. Note that when multiple direction cues from different sonobuoys are received, the direction cue with minimal hop count is deterministically chosen. Algorithm 2 summarizes this beaconing and node state update process.

Algorithm 2. Enhanced beaconing

```
1: procedure BroadcastPeriodicBeacon(node)
   m: a new beacon message
2: if beacon timeout expired then
3:     m.π ← π(node)
4:     m.DF_dir ← DF_dir(node)
5:     m.hop_count ← hop_count(node)
6:     Broadcast m
7:     Set a new timeout
8: end if
9: end procedure
10:
11: procedure ReceiveBeacon(node, m)
12: if m.seq_num > seq_num or
    m.seq_num = seq_num &
    m.hop_count + 1 < hop_count(node) then
13:    NDF_dir(node) ← m.DF_dir
14:    hop_count(node) ← m.hop_count + 1
15:    if π(node) > m.π then
16:        DF_dir(node) ← up
17:    else
18:        DF_dir(node) ← down
19:    end if
20: end if
21: end procedure
```

Fig. 3 shows an example to illustrate Algorithms 1 and 2. The sonobuoy initializes a beacon message after the beacon timer has expired and then broadcasts the beacon message with the sequence number (= 0), depth (= 0), data forwarding direction (= *up*), and hop count (= 0). Node a receives the beacon and finds that it is a new beacon with a higher sequence number and sets its status (e.g., $seq\_num = 0$ with incremented $hop\_count(a) = 1$). By comparing the depth, node a sets $DF\_dir(a)$ as *up*, and $NDF\_dir(a)$ (i.e., the sonobuoy's data forwarding direction) as *up*. Node a will broadcast an updated beacon, and node b will perform a similar procedure, which will be continued. Later, node x receives a beacon message from node b; it then updates $DF\_dir(x)$ as *down* based on the depth difference and

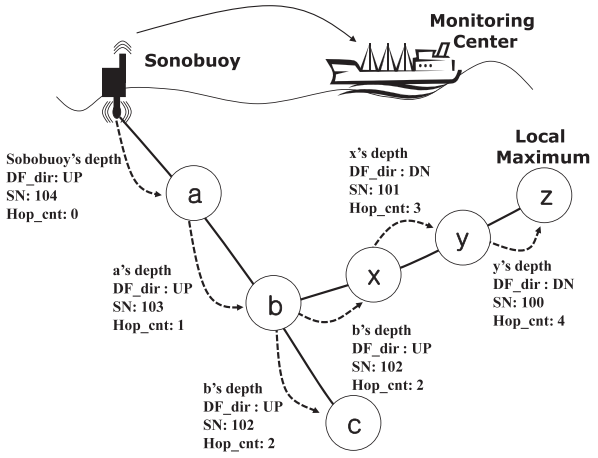


Fig. 3. Enhanced beacon propagation.

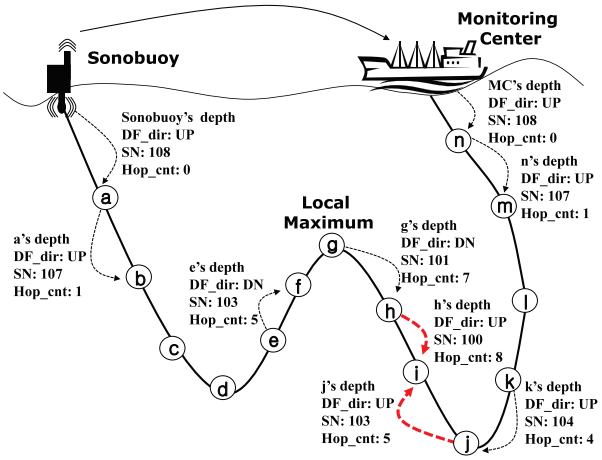


Fig. 4. Beacon receptions in both directions (node i).

$NDF\_dir(x)$ as $DF\_dir(b) = up$. Node x will broadcast an updated beacon message. After these changes are announced via a beacon message, node y receives the beacon message and will maintain $DF\_dir(y)$ and $NDF\_dir(y)$ as *down-down*. On the basis of this beacon propagation and update process, nodes will set up a set of directional trails toward any one of the sonobuoys.

When multiple direction cues from different sonobuoys are received, direction flapping may occur. VAPR uses the sequence number and hop count to prevent such flapping. Whenever a node receives a beacon message with a higher sequence number than its current sequence number, the node simply updates its status based on the received beacon. However, if a node receives a beacon message with the same sequence number, we compare the hop counts, and the data forwarding direction is set toward the node that has a smaller hop count. If a tie occurs, there are two possible cases: a node's current data forwarding direction (i.e., DF_dir) is identical or different (beacons from both shallower and deeper depth levels). The former case of the identical direction can be safely ignored as there will be no impact on the direction setting. For the latter case, we must deterministically use a preferred direction to prevent route flapping; in our scenario, the upward direction is used by default. In Algorithm 2, we omit the details about tie-break in the procedure of RECEIVEBEACON for the sake of brevity. In Fig. 4, for instance, node i receives beacon messages from both directions (from h and j). Node i chooses the forwarding direction toward the closer sonobuoy (*down* in this case) by comparing the hop counts. If both hop counts are the same (a tie), we deterministically set the data forwarding direction as *up*. Note that hop counts are only used for setting up the trails and are not considered when routing data at all—during data forwarding, nodes forward data based solely on the data forwarding and next-hop data forwarding directions, fully exploiting opportunistic directional forwarding, which will be explained in the following section.

4.2 Opportunistic Directional Data Forwarding

Directional data forwarding. In VAPR, nodes forward data packets solely based on the *data forwarding direction* (DF_dir) and *next-hop data forwarding direction* (NDF_dir). Recall that each node sets up the data forwarding direction (either *up* or *down*) that is the opposite direction of the beacon reception direction. If this direction is *up*, we use greedy upward forwarding; otherwise, we use greedy downward forwarding. For instance, in greedy upward/downward forwarding, a node basically forwards a packet to the node whose depth is the shallowest/deepest among its neighbors, respectively.

As the data packet is forwarded upward beneath the concave area of *voids*, a change of data forwarding direction is inevitable. Data forwarding direction alone cannot provide sufficient information to route packets to the destination. In conjunction, we use the next-hop data forwarding direction, which was the predecessor's data forwarding direction during beacon propagation and now becomes the next-hop's data forwarding direction. The forwarding node uses the next-hop's data forwarding direction as an additional *direction constraint* to ensure that routing properly follows the direction trails; i.e., among all neighboring nodes, we only consider the neighboring nodes whose *data forwarding directions* are equal to the *next-hop data forwarding direction* of the current node. As illustrated earlier, there are only four possible cases of data forwarding and next-hop data forwarding direction setting: *up-up*, *down-down*, *down-up*, and *up-down*. The direction changes happen in the latter two cases: from *up* to *down* in the case of *up-down* ($\bigwedge$-shape topology), and from *down* to *up* in the case of *down-up* ($\bigvee$-shape topology).

Consider an example scenario depicted in Fig. 5. In particular, let us take a look at the $\bigvee$-shape topology formed by nodes a, b, and x. Here, node x is a trapped node that eventually delivers packets to the local maximum via greedy upward forwarding. DF_dir and NDF_dir of nodes a and b are *up-up*, whereas those of node x are *dn-up*. Let us say that there is a packet to send in node b. Node b's DF_dir is *up* and will consider nodes whose depth is shallower than that of node b, namely nodes a and x. Since node x's DF_dir (*down*) does not match with that of NDF_dir (*up*), the trapped node x is filtered out, and node a is only considered as a forwarding candidate for local greedy "upward" forwarding.

Enhancement with geoopportunistic forwarding. So far, a packet is greedily forwarded to the node closest to the

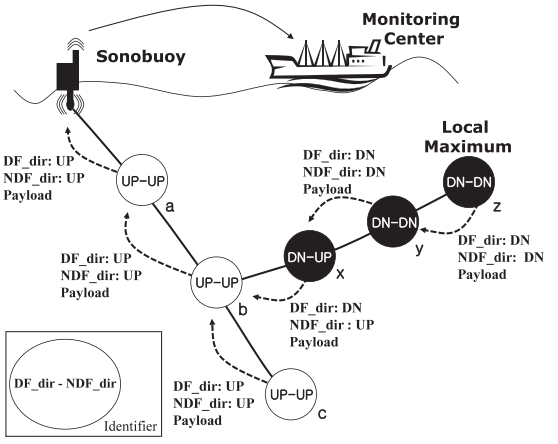


Fig. 5. Directional data forwarding.

destination, in hopes of minimizing the average hop count to the surface. Due to channel fading, however, the farther the transmission range, the higher the attenuation, and the greater the likelihood of packet loss. In VAPR, we consider simultaneous packet receptions by one's neighbors and their ability to opportunistically forward packets by scheduling the set of nodes that have received the packet correctly, which is widely used in geographic routing to improve routing performance under channel fading [17], [33], [26].[2] The key design issue of geo-opportunistic forwarding is the selection of a subset of neighbors that can make the best progress for a given direction, yet without the hidden terminal problem: i.e., when a higher priority node (based on the distance) transmits a packet, other low priority nodes should be able to suppress forwarding to prevent redundant packet transmissions and collisions. Given that finding an optimal set is computationally hard, several heuristics were proposed in the past: a geometric shape (e.g., a triangle/cone [33], [26], a depth-based threshold (e.g., DBR [19]), or greedy clustering (e.g., HydroCast [16]).

In VAPR, we use a simple greedy clustering approach that is superior to the geometric shape or depth-based approaches [16]. To this end, each node requires the knowledge of 2-hop connectivity and neighboring nodes' pairwise distances. Recall that for offline localization we assume that each node measures the pairwise distance [6], and the data are periodically reported to the surface. VAPR takes advantage of such periodic reports to obtain 2-hop neighbor information. We start with a node whose priority is highest (i.e., furthest distance) along the data forwarding direction and choose a group of nodes among its neighbors within the distance $< \beta R$. Here, β is a constant ($\beta < 1$, $\beta = 1/2$ in our design) and R is the acoustic communication range. Then, if other neighbors are left, clustering proceeds to the second highest priority from remaining neighbors and so on, until no nodes are left. After this, each cluster is expanded by including all the additional nodes such that the distance between any two nodes in the cluster is smaller than R. This condition guarantees that nodes in the set can hear each other (i.e., no hidden terminals). We repeat this for all other clusters in turn and find the cluster with the highest expected packet advancement toward the selected direction.

After forwarding the set selection, we need to include the chosen forwarding set in the data packet. To reduce the overhead, we use a Bloom filter, a space efficient membership checking data structure. The membership checking is probabilistic and false positives are possible, but we can bound the probability of false positives by properly adjusting the filter size. In a practical scenario, the set size will be smaller than 15 (in the hemisphere advance zone). Fan et al. [43] showed that a filter size of 150 bits (19B) to represent 15 items has a false positive rate smaller than 1 percent. We can also include sender's depth, and max/min angle information to filter out quite a few of neighboring nodes that are not in the forwarding set. Furthermore, noting that there could be many other packets that have to travel through a certain node, and topology slowly changes over time, we may only need to include the set information in the data packet whenever there is a sufficient change. Thus, the amortized overhead could be much smaller. Algorithm 3 provides a simplified opportunistic directional data forwarding algorithm. The algorithm invokes a function called, $Clustering\_FSS(F, node, data)$ to select possible forwarding nodes based on DF_dir and NDF_dir among its one-hop neighbors and performs clustering to find the best cluster and to return a subset of one-hop nodes that can make the best progress without the hidden terminal problem.

Algorithm 3. Opportunistic Directional Data Forwarding Set Selection

1: **procedure** Directional_FSS($node$, $data$)
2: $F = \phi$ // start with empty set
3: // check all neighbors
4: **for** $n \in$ neighbors(node) **do**
5: // FSS for greedy downward forwarding
6: **if** $DF\_dir(node) = down$
and $\pi(node) \geq n.\pi$
and $n.DF\_dir = data.NDF\_dir$ **then**
7: $F \leftarrow F \cup n$
8: **end if**
9: // FSS for greedy upward forwarding
10: **if** $NDF\_dir(node) = up$
and $\pi(node) \leq n.\pi$
and $n.DF\_dir = data.NDF\_dir$ **then**
11: $F \leftarrow F \cup n$
12: **end if**
13: **end for**
14: // Perform greedy clustering to find the best cluster
15: $C =$ Clustering_FSS(F, node, data)
16: Return C
17: **end procedure**

4.3 Discussion on the Loop Free Property

For the completeness of the routing algorithm, loop freedom in static and mobile networks must be provided. Most ad hoc routing protocols guarantee loop freedom

2. Note that conventional opportunistic routing protocols (also called anypath routing) such as ExOR [14], Least Cost Opportunistic Routing (LCOR) [15] do not use geographic information, but require global topology and link quality information (like link state routing) to find a set of forwarding groups toward the destination; thus, they are more suitable for static wireless mesh or sensor networks.

based on the following observation. Periodic routing request flooding basically builds a reverse path tree toward the source. Route replies will follow the reverse path along which the hop count monotonically decreases. In fact, this simple property ensures a *strict* ordering of feasible distances along successor paths, and thus, loop freedom is guaranteed. For instance, the RREQ tree is formed via the conventional reverse-path flooding techniques of the Ad hoc On Demand Distance Vector (AODV) routing protocol; similarly, the Destination-Sequenced Distance Vector (DSDV) routing protocol periodically performs network-wide flooding with new sequence numbers to update the routing tables. This strict ordering of feasible distances for a given destination is attained by ensuring the Numbered Distance Condition (NDC), as follows [44], [45].

Definition 1 (Numbered Distance Condition). *Node A may accept a route advertisement from neighbor B from destination D and update its routing table independently of other nodes if A has no information about destination D or if either one of the following two conditions is satisfied: $seq\_num_D^A < seq\_num_D^B$ or $seq\_num_D^A = seq\_num_D^B and hop\_count_D^A > hop\_count_D^B$. Here, $seq\_num_D^A$ denotes the sequence number to destination D sent from node A and $hop\_count_D^A$ denotes the hop count to destination D sent from node A.*

Loop-free property of VAPR. While existing routing protocols ensure the NDC property using network wide "instant" flooding, we want to show that the enhanced periodic beaconing in VAPR ensures the NDC property and guarantees the loop-free property. If a network is static, the formal proof is quite straightforward.

We can basically use proof-by-contradiction. For the sake of simplicity, the hop count is used to show the progress to the surface, by assuming that an instance of greedy forwarding has the same effect of decrementing a hop count by one. The proof can also be easily extended to consider the physical distance. During route trail construction, a beacon that carries path information from one of the sonobuoys reaches each node in a connected network. The hop distance monotonically increases along the path. The route from the node to the sonobuoy (return route) follows, by construction, a path with monotonically decreasing hop count. The return route must have the same number of hops as the incoming route. If the return route were shorter, the beacon on the shorter route would have labeled the node. If it were longer, it would have been suppressed by the shorter route, and thus, this cannot happen. By the route trail construction, the return route cannot lead to a dead end. Thus, it must end up either at the sonobuoy that labeled the node, or at another sonobuoy at equal distance.

In mobile networks, existing protocols ensure loop-freedom using either on-demand (e.g., AODV) or periodic (e.g., DSDV) network-wide flooding. A given sequence number will be instantly available throughout the network, and a strict ordering of feasible distance happens—the speed of message propagation is an order of magnitude faster than nodal mobility. However, this means that after some time, the strict ordering may break, and to guarantee the loop-free property in a mobile network, the network must be constantly flooded, which is an expensive process in underwater acoustic networks.

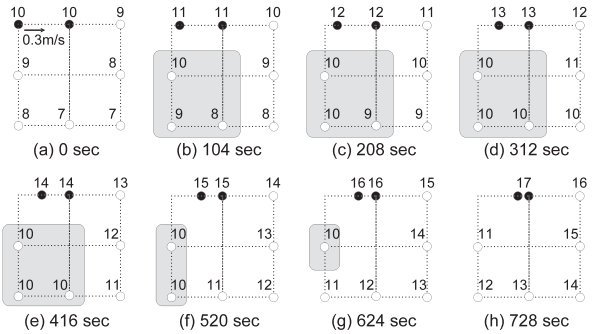


Fig. 6. Beacon propagation with nodal mobility of $0.3\ \mathrm{m/s}$ and beacon interval of 104 s.

Instead of "periodic" instant flooding, VAPR embeds route discovery into the beaconing process. Then the question is how the property of periodic instant flooding can be emulated using the enhanced beaconing process. We note that the flooding interval in traditional routing (e.g., DSDV) is mainly determined by the transmission range and node mobility. If the transmission range is around 250 m, and the relative node speed is 10 m/s, we may want to set the interval smaller than the average time of traveling the transmission range (i.e., 25 s). For instance, in a highly mobile scenario, DSDV is typically configured to run full table updates every 15 s [46]. To illustrate how the periodic beaconing should be configured in VAPR, let us consider the following scenario. Assuming that the maximum distance is K (from the sonobuoys), to ensure the property of "instant" flooding, a given sequence number needs to be propagated within the link lifetime that are mainly dependent on the transmission range and node mobility [47]. The approximate relationship can be represented using the following inequality: K*Beacon Interval $\leq \alpha$ Transmission Range/Nodal Speed (NS) where α is constant. Thus, we have $K * NS \leq \alpha TR/BI$. To summarize, the sequence number propagation speed ($=$ TR/BI) and the maximum depth K are the key factors in determining the loop-free property in VAPR.

For instance, consider a scenario with a nodal speed of $0.3\ \mathrm{m/s}$. Assuming that we have $K = 8$, the sequence number should propagate at the speed of $8 \times 0.3\ \mathrm{m/s}$ ($2.4\ \mathrm{m/s}$). Assuming that the transmission range is $250\ \mathrm{m}$, the beacon interval should be smaller than $250\ \mathrm{m}/(8 \times 0.3\ \mathrm{m/s}) = 104s$. In Fig. 6, we present a 2×2 grid topology in which there are two sonobuoys (black dots) and five regular nodes (white dots), and the side length of the grid is $250\ \mathrm{m}$. In this example, we assume that one sonobuoy moves toward the other sonobuoy at a speed of $0.3\ \mathrm{m/s}$, and that the rest of the nodes are stationary. In the initial state shown in Fig. 6a, the up-to-date sequence number is 10, and both sonobuoys have a synchronized sequence number. In Fig. 6b, the gray area becomes disconnected right after the departure; nodes in that area suffer from transient disconnection. At the same time, the sequence numbered 10 is propagated to the gray area (9, 8, 7 to 10, 9, 8). In Fig. 6c, the lead node no longer receives a new sequence number because it goes out of reach from the left sonobuoy. This

node keeps the sequence number 10 until it receives a newer sequence number. In the meantime, the nodes outside the gray area continue to propagate the sequence numbers in each beacon interval. In Fig. 6d, nodes in the gray area all have the sequence number 10, and they are waiting for the sequence number 11. After four beacon intervals (i.e., at the time mark of 728 s in Fig. 6h), the transient disconnection is resolved, and all nodes have a strict ordering to the sonobuoy on the right.

In VAPR, to reduce the overhead, we aggressively use a larger beacon interval (i.e., by using a smaller value than K, the maximum distance). In the above example (Fig. 6), when we set the K value smaller than 8, it takes a longer time to become loop-free than the case in which K is set to the maximum distance. For instance, if we set $K = 4$, the beacon interval is 204 s (twice the original value). Since it takes seven beacon intervals to receive a new sequence number, the transient instability lasts for 1,456 s.

Fortunately, in practice the effect of route instability can be minimized due to the unique characteristic of underwater sensor networks and VAPR's routing strategies, namely 1) restricted/clustered mobility patterns of underwater sensors (moving along with water current), 2) the multipath nature of opportunistic routing, and 3) beacons sent from multiple sonobuoys. Underwater sensor nodes maintain a fixed depth and move along with the water current. Their mobility patterns are naturally clustered and lead to restricted movement within the cluster. Since sensor nodes are ordered based on their depths, it is likely the case that the distance ordering (hop count) follows the depth order—clustered mobility patterns make deviation of ordering small. In VAPR, any node can maintain distance ordering as long as at least one of the next-hop neighboring nodes receives a newer sequence number (due to opportunistic forwarding). Moreover, it is likely the case that a node will receive beacons from multiple sonobuoys, making more paths toward the surface. In Section 5.2, we further investigate the effect of different beacon intervals on Packet Delivery Ratio (PDR) and energy consumption.

5 Simulations

5.1 Simulation Setup

For acoustic communications, the channel model described in [48] and [49] is implemented in the physical layer of QualNet. The path loss over a distance d for a signal of frequency f, due to large scale fading is given as $A(d, f) = d^k a(f)^d$ where k is the spreading factor and $a(f)$ is the absorption coefficient. The geometry of propagation is described using the spreading factor $(1 \leq k \leq 2)$; for a practical scenario, k is given as 1.5. The absorption coefficient, $a(f)$, is described by Thorp's formula [49]. As in [48] and [50], we use Rayleigh fading to model small scale fading. Unless otherwise mentioned, the transmission power is set to 105 dB re μ Pa. We use a transmission range of 250 m; the data rate is set at 50 Kbps, as in [51]. Our simulations use the CSMA MAC protocol. In CSMA, when the channel is busy, a node waits a back-off period and attempts to sense the carrier again. Every packet transmission is performed through MAC layer broadcasting. For reliability, we implement ARQ at the routing layer as follows for both HydroCast and our proposed routing algorithms. After packet reception, the receiver sends back a short ACK packet. If the sender fails to hear an ACK packet, a data packet is retransmitted; the packet will be dropped after five retransmissions.

We randomly deploy varying numbers of nodes ranging 50 to 550 in a 3D region of size $1{,}500\ \mathrm{m} \times 1{,}500\ \mathrm{m} \times 1{,}500\ \mathrm{m}$. To test routing protocols in a more realistic SEA Swarm scenario, we adopt an extended 3D version of the Meandering Current Mobility Model [52] to model the motility of each sensor node. Unlike most existing sensor node mobility patterns from literature, which assume that each node moves independently of all others, wherein its path vector is determined from an independent realization of a stochastic process, the MCM model considers fluid dynamics whereby the same velocity field advects all nodes. Here, the MCM model considers the effect of meandering subsurface currents (or jet streams) and vortices on the deployed nodes to pattern its path vector.

Meanwhile, additional nodes are deployed in a grid topology on the upper surface of the region (from 1 to 64) to simulate the presence of sonobuoys. Each node measures the distance to its neighbors every 50 s (with random jitters to prevent synchronization) and broadcasts the measured information to its one hop neighbors. Every 50 s, each node reports the sensed data and distance measurements to the surface. The size of a packet is a function of the number of neighbors, and the average packet size in our simulations is less than 200B. We measure packet delivery ratio, average latency per packet, and energy consumption per packet as functions of the number of deployed mobile sensor nodes. The packet delivery ratio of a source is the fraction of the packets delivered; the average latency is the averaged time for every packet to reach any of the sonobuoys on the surface; and the energy consumption is measured in *mWhr* in terms of energy spent per node and per message by each node during the simulation to deliver a packet to the sink. In our simulation, each run lasts 1 hour. Unless otherwise specified, we report an average value of 50 runs with a 95 percent confidence interval.

We have evaluated our proposed routing algorithm against two recent routing protocols: DBR [19] and HydroCast [16]. Recall that DBR greedily forwards packets toward the sea surface using a linear back-off timer proportional to the distance to the destination. This ensures that the nodes closest to the broadcasting node will wait for the nodes closer to the destination that have received the packet to broadcast first. Overhearing the broadcast of the packet by a node closer to the destination serves as an acknowledgment that the packet was forwarded toward the sea surface, and suppresses node transmissions of packets by nodes that are closer to the source, providing an opportunistic forwarding flavor. However, due to lack of an optimized forwarding set selection mechanism, DBR suffers from many redundant transmissions and packet collisions. HydroCast uses a similar linear back-off timer but it calculates an optimal forwarding set based on expected packet advancement [53] and directs the packet to be routed in a general direction relying on opportunistic

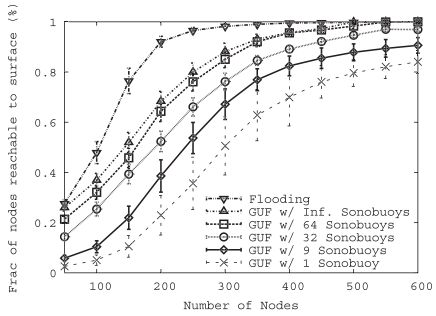


Fig. 7. Fraction of nodes reachable to sonobuoys.

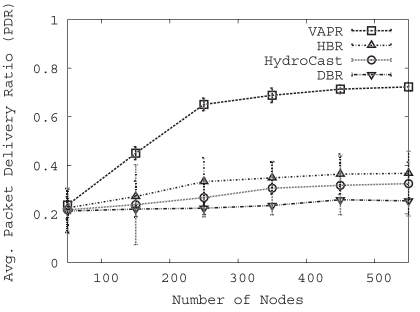


Fig. 8. PDR (*1 sonobuoy scenario*).

packet receptions. If the packet is routed to a trap area, a hop limited ring search is used to build a discovery path along the 2D surface of the convex hull around the void. We evaluate HydroCast with this recovery process. To further evaluate the performance of opportunistic direction forwarding, we additionally compare VAPR with Hop-Based Routing (HBR) that only uses hop-based reachability information to make forwarding decisions: i.e., forwarding a packet to any randomly selected neighboring node whose hop count is smaller than that of the current node. Basically, HBR considers neither physical distance nor opportunistic forwarding; any nodes with the same hop counts are treated equally.

5.2 Simulation Results

We first analyze the network connectivity and its impact on the performance of greedy forwarding under different node and sonobuoy densities. To this end, we perform network-wide flooding from the sonobuoys and measure the fraction of underwater nodes that can reach the surface. We also measure the number with greedy upward forwarding by varying the number of sonobuoys. In the case of network-wide flooding, we do not vary the number of sonobuoys as it is not sensitive to the sonobuoy density. We present the overall results in Fig. 7. The result of flooding shows that when density is low, the fraction of isolated nodes (those that requires performing of route recovery) is significant. The network becomes fully connected when the number of nodes is larger than 400. The results of GUF under varying of sonobuoy density show that the performance of greedy upward forwarding is largely dependent on the sonobuoy density. As the number of sonobuoys increases, the reachability also increases (with diminishing returns). Interestingly, we found that infinite sonobuoys cannot attain the same reachability as is found in flooding due to voids. In fact, the gap between GUF with infinite sonobuoys and flooding represents the fraction of nodes in the trap areas; i.e., these nodes require a route recovery mechanism to reroute packets to sonobuoys. This number is as large as 30 percent of the total number of nodes, especially when the density is low. As the number of sonobuoys decreases, we observe that the gap further increases. Although the network is fully connected, we see that if there is a single sonobuoy, almost 40 percent of the nodes suffer from voids; further, the number of trapped areas decreases as the density increases. The results clearly show the importance of providing a preventive measure for handling voids.

Considering a communication range of 250 m and a 3D ocean cube size of $(1{,}500\ \mathrm{m})^3$, optimal deployments require 91.67 nodes to cover the whole 3D ocean cube. Based on this reachability simulation result, we can claim that a 550 node scenario (i.e., roughly 6 nodes per $(250\ \mathrm{m})^3$ volume) with 64 sonobuoys can provide a reachability ratio of 1 to any of the sonobuoys; moreover, 600 deployed nodes with 1 sonobuoy cannot provide a reachability ratio of 1. To further observe how the number of sonobuoys can affect the protocols' behavior, we deployed a varying number of sonobuoys and provide simulation results for two extreme cases, namely 1 sonobuoy and 64 sonobuoys, under which we can clearly compare each protocol's behavior.

Fig. 8 examines the packet delivery ratio of VAPR, HBR, HydroCast, and DBR with 1 sonobuoy on the surface. The packet delivery ratio of VAPR and HBR outperform those of the rest of the greedy forwarding protocols, namely HydroCast and DBR. This is because these two protocols provide a preventive measure by avoiding trap areas while maintaining only a soft-state in each node. The performance of VAPR is far better than that of HBR due to VAPR's localized opportunistic forwarding. Here, while the number of nodes increases, the PDR does not increase proportionally due to the increased number of retransmissions. More interestingly, the PDR of HydroCast does not increase as the number of deployed nodes increases. HydroCast's recovery process necessitates more frequent ring searches, thereby potentially creating more congestion in the acoustic channel, making it more difficult for HydroCast to deliver packets. This has the effect of diminishing the delivery ratio.

Fig. 9 shows the average energy consumption per message. Energy consumption per message decreases as the number of deployed nodes increases. A higher number of deployed nodes consequently can yield more chances for greedy upward forwarding to succeed without requiring a route recovery process to reach any of the sonobuoys, resulting in less energy consumption per message. DBR's failure of suppressing the redundant packet transmissions causes excessive packet collisions and consumes much more energy than is used in HydroCast, HBR, and VAPR. HydroCast's recovery process near the surface has the effect of diminishing the delivery ratio while increasing the energy costs of HydroCast, in particular with lower node densities, creating more of a distinction from both VAPR and HBR. We note, however, that VAPR and HBR save more energy per packet than does HydroCast, as they do not require packet flooding for route recovery, thereby cutting down on the

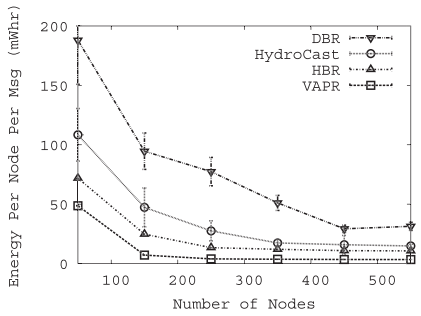


Fig. 9. Energy consumption per message (*1 sonobuoy scenario*).

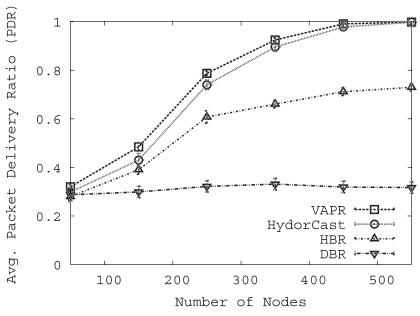


Fig. 11. PDR (*64 sonobuoy scenario*).

likelihood of packet collisions caused by channel congestion and improving the overall energy consumption.

Fig. 10 shows the average latency for all delivered packets. Here, DBR shows the worst performance due to failure of redundant packet suppressions, which causes congestion in the acoustic channel. Both VAPR and HBR outperform HydroCast with route recovery. The improvements are attributed to the clues provided during the beaconing process. Unlike HydroCast, packets generated from trapped nodes no longer require packet rerouting; instead they are routed directly to sonobuoys, without having to be first forwarded to local maxima nodes and then put through a recovery process. Note again that the difference between VAPR and HBR is caused by forwarding set selection granularity. In low density scenarios, opportunistic forwarding can improve the packet delivery ratio; similarly, in high-density scenarios with 1 sonobuoy, opportunistic forwarding reduces the number of packet transmissions, thereby lowering the cochannel interference (and effectively handling the funneling effect).

Fig. 11 examines the packet delivery ratio of VAPR, HBR, HydroCast, and DBR with 64 sonobuoys on the surface. It is possible to see a general trend of positive correlation with node density in VAPR, HBR, and HydroCast but this is not the case with DBR due to the failure of the redundant packet suppressions (which causes congestion in the acoustic channel leading to excessive packet collisions). The packet delivery ratio of HydroCast is similar to that of VAPR; this is because greedy forwarding with a sufficient number of sonobuoys and deployed nodes does not require any route recovery process. HBR cannot effectively handle channel fading, showing a lower PDR than those of Hydrocast and VAPR because HBR does not consider physical distance but hop counts, which means that nodes with the same hop count are treated equally although they have different advancements in terms of physical distances. Finally, this causes a smaller number of nodes to be considered as forwarding nodes. Recall that deployed nodes are moving on the basis of the MCM model (main jet stream speed of 0.3 m/s). Unlike HydroCast, which uses explicit message exchanges to maintain a recovery path, VAPR is a soft-state routing protocol that is more resilient to node mobility and failure.

Fig. 12 shows the energy consumption per message with 64 sonobuoys. The overall trend of the four protocols is quite consistent with that in the previous results, shown in Fig. 9. DBR's failure of redundant packet suppression causes excessive packet collisions. As a result, DBR consumes much more energy than do other protocols. Due to its higher concentration of deployed sonobuoys on the surface, HydroCast does not require a route recovery process. Its performance consequently becomes similar to that of VAPR. We note, however, that HBR's performance does not increase like that of VAPR or HydroCast as the number of deployed nodes increase due to its absence of opportunistic forwarding.

To show the fraction of trapped nodes with respect to the number of sonobuoys, we vary the number of sonobuoys in a range from 1 to 64. As depicted in Fig. 13, the size of the trapped areas depends on the number of sonobuoys on the surface. The lower the number of sonobuoys, the larger the number of trapped nodes. The worst case would be one in which there is only a single sonobuoy. Also note that 3 percent of the nodes are trapped in the 32 and 64 sonobuoys scenarios with 550 deployed nodes, implying that 97 percent of the deployed nodes do not require route recovery to reach any of the sonobuoys (i.e., they are greedy upward forwarding nodes).

Evaluating the beacon interval for the VAPR based on MCM node mobility model is important to show the beacon interval's sensitivity to the speed of node mobility—shorter

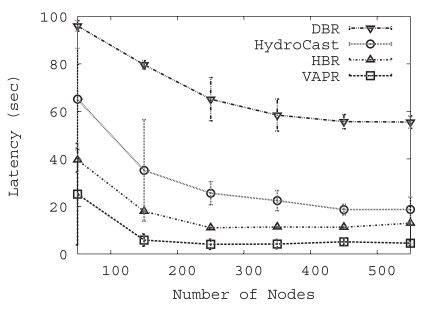


Fig. 10. Avg. latency (*1 sonobuoy scenario*).

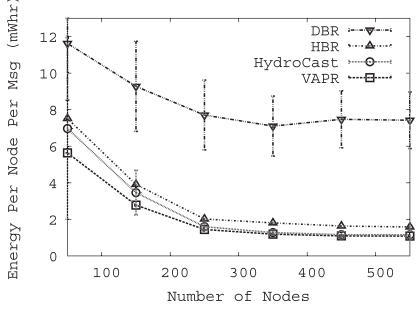


Fig. 12. Energy consumption per message (*64 sonobuoy scenario*).

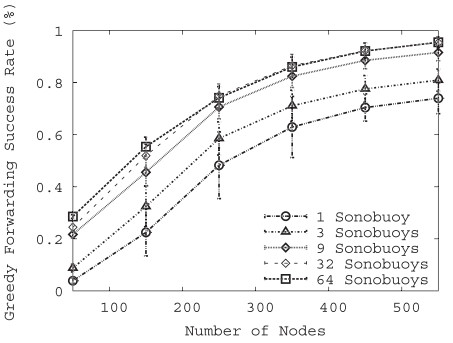


Fig. 13. Fraction of trapped nodes as a function of the number of nodes, under different number of sonobuoys.

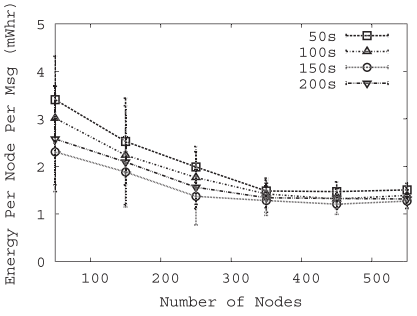


Fig. 15. Effect of different beacon intervals on energy consumption (under MCM mobility).

beacon intervals cause unnecessary overhead while longer beacon intervals cause stale routing information amongst deployed nodes. In Figs. 14 and 15, we show the packet delivery ratio and the average energy consumption per message for VAPR with different beacon intervals of 50 s, 100 s, 150 s, and 200 s with MCM node mobility (0.3 m/s). All intervals show positive correlation with node density. However, it can be seen that the beacon interval of 150 s shows the best results. As depicted in Fig. 14, a beacon interval of 50 s shows the best packet delivery ratio in low densities but saturates earlier than other intervals due to its frequent beacon message generation. However, the beacon interval of 150 s shows stable and desirable performance regarding packet delivery ratio. As depicted in Fig. 15, energy consumption of the 150 s beacon interval shows the best energy savings. It is noteworthy that the 200 s beacon interval shows degraded energy performance compared to that of the 150 s beacon interval in both low and high node densities. As beacon intervals become longer, the routing clues become stale. As a result, beacons provide less accurate routing information, which increases the energy consumption per message necessary to route a message to the sonobuoys.

6 Discussion

As illustrated earlier, one of the key design issues of opportunistic routing is the selection of a subset of neighbors that can make the best progress toward the destination without the hidden terminal problem. The major drawback of existing opportunistic routing such as ExOR [14] and LCOR [15] is that it requires global topology and link quality information (like link state routing) to find a set of forwarding groups toward the destination. Due to the protocol overhead, this approach is less suitable for mobile underwater sensor networks. An alternative to this approach is to augment existing routing protocols such as geographic routing and hop-based routing with *localized opportunistic forwarding*; i.e., a set of neighboring nodes that have shorter distance or hop-count than the current node can jointly forward a packet to make better progress toward the destination. In general, geographic routing can better exploit localized opportunistic forwarding because the expected number of candidate nodes in geographic routing would be much greater than that in hop-based routing. In geographic routing, any node whose distance toward the destination is smaller than that of the current node (or any nodes in the advance zone) is considered, whereas in hop-based routing, any node whose hop-count toward the destination is smaller than that of the current node is only considered.[3] When comparing these approaches, we observe that a significant fraction of the candidate nodes in geographic routing may have the same hop count in hop-based routing. For example, assuming that the current forwarding node has five neighboring nodes in the advance zone of geographic routing, it is possible that only one node has lower hop count in hop-based routing (i.e., 5 versus 1). This argument justifies our design choice of applying localized opportunistic forwarding to pressure routing, a specialized geographic routing scenario. In this paper, we leave the performance comparison of these approaches as part of future work.

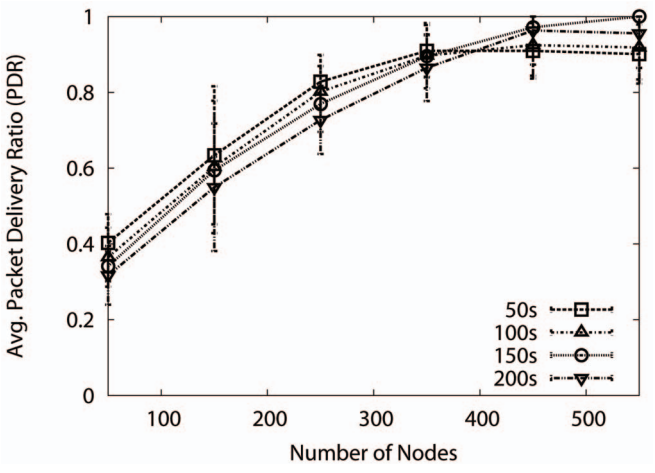


Fig. 14. Effect of different beacon intervals on PDR (under MCM mobility).

7 Conclusion

We investigated pressure routing in underwater mobile sensor networks and have proposed VAPR, a simple and robust soft-state protocol. VAPR exploits periodic beaconing to build directional trails toward the surface and features greedy opportunistic directional forwarding for

3. As in VAPR, hop-based routing with opportunistic forwarding also requires efficient forwarding set selection methods that choose a subset of neighbors that make the best progress toward the destination, yet without the hidden terminal problem. One simple way would be modifying the greedy clustering method of HydroCast [16] (e.g., just finding the cluster with the largest number of nodes).

packet delivery. We provided a detailed discussion on the loop free property of VAPR and showed that the sequence number propagation speed and the maximum depth are the key factors of ensuring loop-freedom in mobile networks. Our extensive simulations showed that VAPR outperforms existing schemes by significantly lowering the frequency of recovery fallbacks and by effectively handling node mobility.

REFERENCES

[1] I.F. Akyildiz, D. Pompili, and T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges," *Ad Hoc Networks,* vol. 3, no. 3, pp. 257-279, 2005.

[2] J. Kong, J.-H. Cui, D. Wu, and M. Gerla, "Building Underwater Ad-Hoc Networks and Sensor Networks for Large Scale Real-Time Aquatic Applications," *Proc. IEEE Military Comm. Conf. (MILCOM),* 2005.

[3] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data Collection, Storage, and Retrieval with an Underwater Sensor Network," *Proc. Third Int'l Conf. Embedded Networked Sensor Systems (Sensys),* 2005.

[4] J. Jaffe and C. Schurgers, "Sensor Networks of Freely Drifting Autonomous Underwater Explorers," *Proc. First ACM Int'l Workshop Underwater Networks (WUWNet),* 2006.

[5] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A Decentralized Network Coordinate System," *Proc. ACM SIGCOMM,* 2004.

[6] V. Chandrasekhar, Y.S. Choo, and H.V. Ee, "Localization in Underwater Sensor Networks—Survey and Challenges," *Proc. First ACM Int'l Workshop Underwater Networks (WUWNet),* 2006.

[7] M. Stojanovic, "On the Relationship Between Capacity and Distance in an Underwater Acoustic Communication Channel," *ACM SIGMOBILE Mobile Computing and Comm. Rev.,* vol. 11, pp. 34-43, 2007.

[8] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball, "The WHOI Micro-Modem: An Acoustic Communications and Navigation System for Multiple Platforms," *Proc. IEEE OCEANS Conf.,* 2005.

[9] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," IETF RFC 3626, 2003.

[10] C.E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *Proc. ACM SIGCOMM,* 1994.

[11] C.E. Perkins and E.M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," *Proc. IEEE Workshop Mobile Computing Systems and Applications,* 1997.

[12] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing,* pp. 153-181, 1996.

[13] V.D. Park and J.P. Macker, "Anycast Routing for Mobile Services," *Proc. Conf. Information Science and Systems,* 1999.

[14] S. Biswas and R. Morris, "Opportunistic Routing in Multi-Hop Wireless Networks," *Proc. ACM SIGCOMM,* 2005.

[15] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Least-Cost Opportunistic Routing," *Allerton,* 2007.

[16] U. Lee, P. Wang, Y. Noh, L.F.M. Vieira, M. Gerla, and J.-H. Cui, "Pressure Routing for Underwater Sensor Networks," *Proc. IEEE INFOCOM,* 2010.

[17] M. Zorzi and R.R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and Latency Performance," *IEEE Trans. Mobile Computing,* vol. 2, no. 4, pp. 349-365, Oct.-Dec. 2003.

[18] B. Jalving, "Depth Accuracy in Seabed Mapping with Underwater Vehicles," *Proc. MTS/IEEE Riding the Crest into the 21st Century (OCEANS),* 1999.

[19] H. Yan, Z. Shi, and J.-H. Cui, "DBR: Depth-Based Routing for Underwater Sensor Networks," *Proc. Seventh Int'l IFIP-TC6 Networking Conf. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet (Networking),* 2008.

[20] S. Durocher, D. Kirkpatrick, and L. Narayanan, "On Routing with Guaranteed Delivery in Three-Dimensional Ad Hoc Wireless Networks," *Wireless Networks,* vol. 16, no. 1, pp. 227-235, Jan. 2010.

[21] R. Flury and R. Wattenhofer, "Randomized 3D Geographic Routing," *Proc. IEEE INFOCOM,* 2008.

[22] Y. Noh, P. Wang, U. Lee, and M. Gerla, "VAPR: Void Aware Pressure Routing Protocol," *WUWNet Work-In-Progress Poster,* 2010.

[23] D. Pompili, T. Melodia, and I.F. Akyildiz, "Routing Algorithms for Delay-Insensitive and Delay-Sensitive Applications in Underwater Sensor Networks," *Proc. ACM MobiCom,* 2006.

[24] P. Xie, J.-H. Cui, and L. Lao, "VBF: Vector-Based Forwarding Protocol for Underwater Sensor Networks," *Networking Technologies, Services, and Protocols; Performance of Computer and Comm. Networks; Mobile and Wireless Comm. Systems,* vol. 3976, pp. 1216-1221, 2006.

[25] N. Nicolaou, A. See, P. Xie, J.-H. Cui, and D. Maggiorini, "Improving the Robustness of Location-Based Routing for Underwater Sensor Networks," *Proc. IEEE OCEANS Conf.,* 2007.

[26] J.M. Jornet, M. Stojanovic, and M. Zorzi, "Focused Beam Routing Protocol for Underwater Acoustic Networks," *Proc. Third ACM Int'l Workshop Underwater Networks (WUWNet),* Sept. 2008.

[27] L. Vieira, U. Lee, and M. Gerla, "Phero-Trail: A Bio-Inspired Location Service for Mobile Underwater Sensor Networks," *IEEE J. Selected Areas in Comm.,* vol. 28, no. 4, pp. 553-563, May 2010.

[28] P. Casari and A.F. Harris, "Energy-Efficient Reliable Broadcast in Underwater Acoustic Networks," *Proc. Second Workshop Underwater Networks (WUWNet),* 2007.

[29] P. Casari, M. Rossi, and M. Zorzi, "Towards Optimal Broadcasting Policies for HARQ Based on Fountain Codes in Underwater Networks," *Proc. Fifth Ann. Conf. Wireless on Demand Network Systems and Services (WONS),* 2008.

[30] P. Nicopolitidis, G. Papadimitriou, and A. Pomportsis, "Adaptive Data Broadcasting in Underwater Wireless Networks," *IEEE J. Oceanic Eng.,* vol. 35, no. 3, pp. 623-634, July 2010.

[31] P. Casari and M. Zorzi, "Protocol Design Issues in Underwater Acoustic Networks," *Computer Comm.,* vol. 34, no. 17, pp. 2013-2025, 2011.

[32] M. Ayaz, I. Baig, A. Azween, and F. Ibrahima, "A Survey on Routing Techniques in Underwater Wireless Sensor Networks," *J. Network and Computer Applications,* vol. 34, pp. 1908-1927, 2011.

[33] H. Füßler, J. Widmer, M. Käsemann, M. Mauve, and H. Hartenstein, "Contention-Based Forwarding for Mobile Ad-Hoc Networks," *Ad Hoc Networks,* vol. 1, no. 4, pp. 351-369, Nov. 2003.

[34] A. Savvides, C.-C. Han, and M.B. Strivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors," *Proc. ACM MobiCom,* 2001.

[35] P. Biswas and Y. Ye, "Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization," *Proc. Third Int'l Symp. Information Processing in Sensor Networks (IPSN),* 2004.

[36] J. Opatrny, A. Abdallah, and T. Fevens, "Randomized 3D Position-Based Routing Algorithms for Ad-Hoc Networks," *Proc. Third Ann. Int'l Conf. Mobile and Ubiquitous Systems: Networking Services,* 2006.

[37] W. Zeng, R. Sarkar, F. Luo, X. Gu, and J. Gao, "Resilient Routing for Sensor Networks Using Hyperbolic Embedding of Universal Covering Space," *Proc. IEEE INFOCOM,* 2010.

[38] T. He, J.A. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks," *Proc. 23rd Int'l Conf. Distributed Computing Systems (ICDCS),* 2003.

[39] B. Leong, B. Liskov, and R. Morris, "Geographic Routing Without Planarization," *Proc. Third Conf. Networked Systems Design and Implementation (NSDI),* 2006.

[40] K. Liu and N.B. Abu-Ghazaleh, "Virtual Coordinate Backtracking for Void Traversal in Geographic Routing," *Proc. Fifth Int'l Conf. Ad-Hoc, Mobile, and Wireless Networks (ADHOC-NOW),* 2006.

[41] B. Zhou, Y.Z. Lee, M. Gerla, and F. de Rango, "Geo-LANMAR: A Scalable Routing Protocol for Ad Hoc Networks with Group Motion," *Wireless Comm. and Mobile Computing,* vol. 6, pp. 989-1002, 2006.

[42] J. Zhou, Y. Chen, B. Leong, and P.S. Sundaramoorthy, "Practical 3D Geographic Routing for Wireless Sensor Networks," *Proc. Eighth ACM Conf. Embedded Networked Sensor Systems (SenSys),* 2010.

[43] L. Fan, P. Cao, and J. Almeida, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol," *Proc. ACM SIGCOMM,* 1998.

[44] J.J. Garcia-Luna-Aceves, M. Mosko, and C.E. Perkins, "A New Approach to On-Demand Loop-Free Routing in Ad Hoc Networks," *Proc. 22nd Ann. Symp. Principles of Distributed Computing (PODC),* 2003.

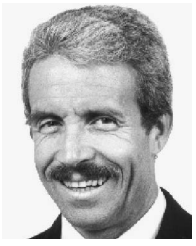
[45] J. Garcia-Luna-Aceves and H. Rangarajan, "A New Framework for Loop-Free On-Demand Routing Using Destination Sequence Numbers," *Proc. Int'l Conf. Mobile Ad-hoc and Sensor Systems (MASS)*, 2004.
[46] R. Boppana and S. Konduru, "An Adaptive Distance Vector Routing Algorithm for Mobile, Ad Hoc Networks," *Proc. IEEE INFOCOM*, 2001.
[47] P. Samar and S.B. Wicker, "On the Behavior of Communication Links of a Node in a Multi-Hop Mobile Environment," *Proc. ACM MobiHoc*, 2004.
[48] M. Stojanovic, "On the Relationship Between Capacity and Distance in an Underwater Acoustic Communication Channel," *Proc. First ACM Int'l Workshop Underwater Networks (WUWNet)*, 2006.
[49] L.M. Brekhovskikh and Y. Lysanov, *Fundamentals of Ocean Acoustics (Modern Acoustics and Signal Processing)*, third ed. Springer, 2003.
[50] C. Carbonelli and U. Mitra, "Cooperative Multihop Communication for Underwater Acoustic Networks," *Proc. First ACM Int'l Workshop Underwater Networks (WUWNet)*, 2006.
[51] Z. Zhou and J.-H. Cui, "Energy Efficient Multi-Path Communication for Time-Critical Applications in Underwater Sensor Networks," *Proc. ACM MobiHoc*, 2008.
[52] A. Caruso, F. Paparella, L. Vieira, M. Erol, and M. Gerla, "The Meandering Current Mobility Model and its Impact on Underwater Mobile Sensor Networks," *Proc. IEEE INFOCOM*, 2008.
[53] K. Zeng, W. Lou, J. Yang, and D. Brown, "On Geographic Collaborative Forwarding in Wireless Ad Hoc and Sensor Networks," *Proc. Int'l Conf. Wireless Algorithms, Systems and Applications (WASA)*, 2007.

Youngtae Noh received the BS degree in computer science from Chosun University in 2005 and the MS degree in information and communication from Gwangju Institute of Science Technology (GIST) in 2007. Currently, he is working toward the PhD degree in computer science at the University of California, Los Angeles (UCLA). His research areas include wireless networking, future Internet, and mobile computing. He is a member of the IEEE.

Uichin Lee received the BS degree in computer engineering from Chonbuk National University in 2001, the MS degree in computer science from KAIST in 2003, and the PhD degree in computer science from the University of California, Los Angeles (UCLA) in 2008. He is an assistant professor in the Department of Knowledge Service Engineering at the Korea Advanced Institute of Science and Technology (KAIST). Before joining KAIST, he was a member of the technical staff at Bell Laboratories, Alcatel-Lucent, until 2010. His research interests include distributed systems and mobile/pervasive computing. He is a member of the IEEE.

Paul Wang received the BS/BA degrees in computer science and economics from the University of California, San Diego (UCSD) in 2006 and the MS degree in computer science from the University of California, Los Angeles (UCLA) in 2010. He is a software systems engineer at the Jet Propulsion Laboratory (JPL) run by the California Institute of Technology (CIT) on behalf of NASA. His research interests include distributed systems, ad hoc and challenged networks, mobile computing, and large-scale database systems. He is a member of the IEEE.

Brian Sung Chul Choi received the BS degree in electrical engineering and computer sciences from UC Berkeley in 2006, and the PhD degree in computer science from the University of California, Los Angeles (UCLA) in 2011. He is a software engineer at Google, Inc. He is a member of the IEEE.



Mario Gerla received the engineering degree from Politecnico di Milano, Italy and the PhD degree from UCLA. He is a professor in the computer science at UCLA. At UCLA, he was a part of the team that developed the early ARPANET protocols under the guidance of Prof. Leonard Kleinrock. At Network Analysis Corporation, New York, from 1973 to 1976, he helped transfer ARPANET technology to Government and Commercial Networks. He joined the UCLA Faculty in 1976. At UCLA he has designed and implemented network protocols including ad hoc wireless clustering, multicast (ODMRP and CodeCast) and Internet transport (TCP Westwood). He has lead the $12M, 6 year ONR MINUTEMAN project, designing the next generation scalable airborne Internet for tactical and homeland defense scenarios. He is now leading two advanced wireless network projects under ARMY and IBM funding. His team is developing a Vehicular Testbed for safe navigation, urban sensing and intelligent transport. A parallel research activity explores personal communications for cooperative, networked medical monitoring He is a fellow of the IEEE. (see www.cs.ucla.edu/NRL for recent publications).