Revisiting Source Routing for Underwater Networking: The SUN Protocol

Giovanni Toso, Riccardo Masiero, Paolo Casari, *Senior Member, IEEE*,
Maksym Komar, Oleksiy Kebkal, Michele Zorzi, *Fellow, IEEE*

***Abstract*—With respect to other routing paradigms, source routing has received comparatively less attention in the underwater acoustic networking domain. The most likely causes of this lack of momentum are the high overhead caused by route discovery and maintenance in typical implementations of the source routing paradigm (e.g., Dynamic Source Routing, DSR) in terrestrial radio networks.**

In this paper, we revert this view and argue that Source Routing can in fact be a reliable and convenient routing paradigm in underwater networks, when properly implemented and tailored to the peculiarities of underwater acoustic channels. Our scheme, named SUN, successfully recasts the source routing approach by introducing a number of new features, that improve the routing performance especially in the presence of unstable network links and mobile nodes. SUN is scenario-independent by design: this means that it can work in any connected topology, and does not need any side information (such as the node location and depth, or the channel state) in order to operate correctly.

We evaluate the performance of SUN by means of simulations using the DESERT Underwater framework. Our results show that SUN correctly manages routing in both static and mobile networks, and that in some scenarios it even achieves better performance than a competing flooding-based approach. We also test the performance of SUN in a thorough experimental campaign involving 6 nodes and carried out in a lake near Berlin. From these results, we conclude that SUN, and the source routing paradigm in general, are in fact feasible options for general-purpose routing in underwater acoustic networks.

***Index Terms*—Underwater networks; source routing; SUN protocol; DESERT Underwater; simulations; lake experiment.**

I. Introduction

UNDERWATER Acoustic Networks (UANs) are perceived as an increasingly feasible approach for several applications, such as oceanographic data collection, water monitoring, offshore exploration, disaster prevention and assisted navigation. Depending on the specific scenario, UANs may consist of a variable number of entities, both mobile and static. These include sensor nodes, Autonomous Underwater Vehicles (AUV), buoys, and ships, that can collaborate in order to carry out a task in a given area. Moreover, some nodes can be anchored, while others can be mobile (e.g., drifters and floaters).

Part of this work has been presented at the MTS/IEEE OCEANS 2012 conference, Hamptons Road, VA, USA [1].

P. Casari (e-mail: paolo.casari@imdea.org) is with IMDEA Networks, 28918 Madrid, Spain. M. Komar (e-mail: maks@evologics.de) and O. Kebkal (email: lesha@evologics.de) are with EvoLogics GmbH, 13355 Berlin, Germany. M. Zorzi (e-mail: zorzi@dei.unipd.it) is with the Department of Information Engineering, University of Padova, 35131 Padova, Italy. This work has been carried out in part when G. Toso and R. Masiero also were at the Department of Information Engineering, University of Padova, Italy.

Whenever operations over a large area are required, such networks become inherently multihop. This is partly due to the limited range of underwater acoustic transmission equipment, and partly to the fact that shorter acoustic links are characterized by a larger bandwidth, and less energy is required to communicate over them [2]. In such multihop scenarios, the nodes must autonomously organize into a network and find multihop routes to deliver data to their intended destinations. This mechanism is usually delegated to a routing protocol. There are several types of solutions to the routing problem in underwater networks [3]. These include dynamic vs. static routing, source routing vs. hop-by-hop relay selection, proactive vs. reactive and distributed vs. centralized protocols. Each approach has its own pros and cons. In this paper, we argue that source routing is a feasible approach for generic underwater networks, despite the comparatively smaller amount of attention it has received in the body of research carried out so far. Our objective is therefore to recast source routing in underwater networks by designing a protocol that is specifically tailored to the underwater environment. To this end, we propose a reactive Source routing protocol for Underwater acoustic Networks (SUN). SUN has been developed with the characteristics of UANs in mind, and was designed to be applicable to generic underwater network topologies (i.e., it is not tailored to a specific scenario). Moreover, it is dynamic by design, hence capable to adapt to substantial changes in the network topology, as caused for instance by significant channel variations over time, or by the movement, departure or appearance of network nodes. Finally, in the design of SUN we make no assumptions regarding the structure of the network topology, the knowledge of the network links, the location of the nodes, or the channel state between them.

The paper is organized as follows: Section II covers the related literature; Section III describes the SUN protocol; Section IV presents simulation results used to validate the design of SUN; Section V discusses the results of our lake experiments where we tested SUN in a real-world environment; finally, Section VI concludes the paper.

II. Related Work

Different approaches have been applied to the design of routing protocols for underwater networks. These include flooding-based, opportunistic, geographic, multipath and cluster-based protocols, along with some hybrid schemes [4]–[6]. The flooding approach is the simplest and requires little if any knowledge of the network. Flooding-based protocols prescribe that every previously unseen packets received by a node be forwarded to all of the node's neighbors. The

main disadvantage is that flooding originates many duplicate packets, potentially leading to broadcast storms. In turn, this translates into a high energy consumption per packet delivered to its intended destination, and can lead replica transmissions to interfere with one another. However, some attempts to reduce the number of retransmissions in specific underwater scenarios [7] and in generic networks [8]–[10] have also been presented. Some flooding-based routing protocols such as SBR-DLP [11] and DFR [12] also require location information.

Vector Based Forwarding (VBF) [13] and its adaptive extension [14] prescribe that packet delivery be guided by the vector joining the source (or the current relay in [14]) of a packet to its destination: only the nodes within a specific range of this vector can forward packets. The Focused Beam Routing (FBR) protocol [15] selects relays hop-by-hop via an RTS/CTS exchange. FBR tries to minimize the energy consumption over a multihop path by progressively increasing the transmission power of RTSs until a possible relay answers the request, and is suitable for both mobile and static networks. Depth-Based Routing (DBR) for Underwater Sensor Networks [16] is a greedy protocol inspired to VBF that requires only local depth information. While DBR is shown to achieve very high packet delivery ratios in dense networks, the protocol is constrained to a scenario where the nodes must send data to surface-located sinks. Alternatively, the depth-controlled routing (DCR) [17] protocol proposed to leverage the ability of the nodes to change their depth in order to reduce the probability that local minima are incurred in geographic routing protocols for underwater networks. Hop-by-Hop Dynamic Addressing Based (H2-DAB) routing [18] takes advantage of a multi-sink routing architecture. Like DBR, it does not require any location information, but is also constrained to bottom-to-surface routing. A hop-by-hop relay selection approach is also taken by the CARP protocol, presented in [19]. After creating a converge-casting tree by propagating a beacon downstream from the sink, CARP exploits a local handshaking procedure and several relay quality metrics to select optimal relays at each hop. Simulation results show that CARP outperforms other handshake-based solutions such as FBR and DBR.

Void-Aware Pressure Routing (VAPR) [20] routes traffic from sensors placed deeper in the water column towards one or more buoys on the surface. This is achieved by having the buoys propagate beacon signals towards sensors located in deeper waters. The trails thus established can be traveled towards the surface through local relaying decisions that rely on pressure gauges in order to choose relays that offer progress towards the surface. A review of additional protocols that also require beaconing to achieve routing in UANs is provided in [21]. A distributed relaying scheme is proposed in [22] to achieve near-optimal routing in multi-modal underwater networks, where each node may have more than one physical layer technology through which to forward data. Opportunistic routing concepts are explored in [23], which considers the use of duty cycling to save energy in high-density underwater networks, and in [24], where routing is carried out in a 3D network by balancing the energy consumption across the nodes while keeping the delivery ratio low and the success ratio sufficiently high. The approach in [25] applies fuzzy logic to multi-objective optimization in order to achieve low energy consumption and high delivery ratio. General design guidelines for opportunistic routing are provided in [26].

A different approach is taken by multipath routing, where multiple paths are established from the source to the destination of a given transmission. This can lead to higher reliability and robustness. Two examples of this approach are the schemes in [27] and [28]. In [27], multiple sinks are connected to each other through high speed links (i.e., fiber-optic cables) and multiple paths are discovered during an initialization phase. This solution has the advantage that it does not require location information, but it is constrained by the need to connect all sinks via cables. The approach in [28] is based on the concept of virtual-circuit routing, where connections are established before communicating. The disadvantage of this protocol is that there must exist a centralized manager with complete knowledge of the network. In [29], the authors design a protocol that keeps multipath routes between the same source and destination as node-disjoint as possible, in order to reduce cross-path interference.

Cluster-based protocols define the roles of head and member nodes, where heads collect data from the members in their cluster, and transmit such data to a sink. Two examples are Hydraulic Pressure Based Anycast Routing (HydroCast) [30] and Distributed Underwater Clustering Scheme (DUCS) [31]. In DUCS, clusters are created among nodes, and each cluster elects its head. Periodic re-clustering allows energy consumption balance. A disadvantage of DUCS is that it requires full knowledge of the network and suffers from low throughput in sparse networks. HydroCast, instead, requires a depth sensor like DBR.

Other approaches cannot be strictly classified as vector-based, flooding-based or multipath-based. For example, the Information Carrying based Routing Protocol (ICRP) [32] works as follows: if a node does not have a path to a sink it sends the message in broadcast. When a sink receives a message from a node for the first time, it sends back a route establishment packet to the source, and for the next transmissions the source will use this path. ICRP does not require any location information and works in both static and mobile networks. A concept similar to ICRP is also employed in the GUWMANET protocol [33]. Some attempts to adapt the AODV protocol from the terrestrial radio world to underwater scenarios were performed in [34], where the authors proposed AODV-B1, a reactive routing protocol with low overhead. In contrast to AODV, the routes created are unidirectional, in order to reduce the overhead. Moreover, a route is removed only upon the receipt of error notification packets.

Dynamic Source Routing (DSR) [35] is a routing protocol designed for mobile, ad hoc, wireless radio networks, and would suffer from the large latency of the underwater acoustic environment. In addition, the rate at which underwater topologies may change in the presence of channel quality fluctuations is likely to be high compared to the delay imposed by acoustic communications, and DSR is expected to react too slowly to these changes. The Location-Aware Source Routing (LASR) [36] protocol starts from these considerations. It

employs a link quality metric in place of DSR's hop-count metric, which makes it possible to outperform DSR when the network topology changes rapidly. However, the design of LASR assumes location awareness, requires symmetric and bidirectional links, and prescribes that channel access be administered through a TDMA scheme, which requires strict synchronization and low clock drift. These assumptions are rarely verified in underwater networks. DSR also inspires the protocol proposed in this paper. A detailed discussion about the differences and similarities between SUN and DSR is provided in Section III-F.

The literature review presented in this section suggests that, with very few exceptions, most protocols designed for underwater networks assume one or more of the following: availability of location information; static network topology; availability of specific hardware (e.g., a depth gauge); specific network configuration (e.g., buoys floating on the surface, nodes anchored to the bottom); static and known sinks; wired connections between sinks; or a centralized network manager. Conversely, SUN is designed to achieve good performance in terms of energy efficiency, delivery ratio and throughput, without requiring location information, depth knowledge, or any other among the assumptions above to do so. In addition, it works in networks with both fixed and mobile nodes, and does not require continuous interaction with a centralized control unit. The next section presents the design of the SUN protocol.

III. Description of SUN

A. Main idea

SUN can be broadly described as a reactive, source routing-based protocol, with several additional aspects that have been designed to achieve good performance in the presence of typical underwater acoustic channels. SUN is reactive in order not to waste the limited acoustic bandwidth with proactive route discovery traffic. Moreover, the source routing mechanism gives the source the authority to decide on the route to be followed, avoiding the inherent delays and control signaling of a hop-by-hop relay selection process.

SUN assumes that the underwater network nodes have to deliver some data traffic to any of a set of final receivers, or *sinks*. This assumption is formalized by the separation of the network into two entities: sinks and nodes. Besides receiving `Data` packets, the only other task of a sink is to periodically send one-hop `Probe` messages, in order to notify their own presence to the nodes within their communications range. Instead, other nodes can send `Data` packets, send `Path Request` packets and answer them, relay `Data` packets on behalf of other nodes, as well as notify broken routes.

For improved efficiency, SUN is designed as a cross-layer protocol. In particular, SUN internally buffers both the packets to be transmitted and those received from the lower layers of the protocol stack. A buffering system within the network layer yields several advantages: it makes it possible to store specific `Data` packets, and optionally to decide which packets should be saved or dropped if buffer overflows occur. When a node does not know a valid path to the sink, a buffer gives the possibility to store the packets, send a `Path Request`, wait for an answer, and finally fill the header of the packet with a valid route. Another cross-layer feature implemented in SUN is a Stop-and-Wait Automatic Repeat reQuest (ARQ) mechanism. Performing ARQ at the network level without delegating it to the link control layer makes it possible to perform basic topology control (detecting unreliable paths, congested links, and nodes movement leading to link breakage and creation of new links), without implementing specific control features at the routing layer. In turn this reduces the overhead, and saves the energy that would be otherwise required to explicitly probe the links: both aspects are extremely important in underwater scenarios. These two functions and their integration in SUN are detailed in Section III-B.

B. The SUN algorithm

The routing procedure in SUN works as follows. Every sink periodically sends a `Probe`; if a node receives the message, it understands to be a one-hop neighbor of the sink.[1] The neighbors of the sink have hop count 1 and are termed "end nodes" in the following. These nodes will answer `Path Requests` and will relay packets directed to the sink. A node will be an end node only for a pre-defined period; if it does not receive any further `Probe` from the sink during this period, it will assume that the channel toward the sink has become unreliable, and will release its role.

When the routing module receives a packet from the upper layers, it stores the packet in a buffer. If the buffer is full, the node will drop new packets until at least one slot of the buffer becomes free. The size of the buffer can be configured for each node. An agent checks the buffer periodically, and if any packets are found, they are served according to a First-In-First-Out (FIFO) policy. The behavior of the agent depends on the hop count of the node:

- end nodes (hop count equal to 1) are directly connected to the sink; therefore, the agent initializes the packet, in case it was not previously initialized, and forwards it directly to the sink;[2]
- the nodes with hop count greater than 1 have a valid path to the sink, which can be reached via multihop relaying. As in the previous case, the agent initializes the packet and forwards it to the next hop;
- the nodes with a hop count of 0 do not have a valid path to the sink: therefore, they keep the packet stored in the buffer and start the `Path Establishment` procedure described below.

We note that a fresh initialization is performed every time the packet is read from the buffer, in order to exploit the freshest available path by writing it in the packet's header.

[1]This is rigorously true if the channel can be assumed to be symmetric. Such a condition is not necessarily observed in reality; however, the quality of the return channel from the node to the sink will likely be also good in practice, if the nodes discard the `Probes` received with a signal-to-noise ratio below some threshold. A threshold of 15 dB ensured a return channel with good quality in all simulations.

[2]The "initialization" is the process of creating the packet header: it involves the specification, among other fields, of the full sequence of hops that the packet should traverse towards the sink: for an end node, this means sending the packet directly to the sink.

When a node sends a Data packet to the lower layers, it increases the number of transmission attempts for this packet; additionally, it waits for an acknowledgment packet (ACK) and correspondingly starts an ACK timeout. If a node receives an ACK related to the first packet in the buffer, the node will assume that the ACK sender correctly received that Data packet, and will remove it from the buffer. For each packet a maximum number of retransmissions is set. Upon reaching this number, the current node will consider the link to the next hop (as read from the packet header) unreliable, and:

- it will create a PathError packet, in order to notify the source of the unavailable path;
- if the route in the header of the packet is the same known by the current node, it will also delete its own routing entry;
- it will remove the packet from the buffer.

C. Path Establishment Algorithm

When no valid route toward the destination is known, a node starts the discovery process by flooding a Path Establishment packet with a Request flag (PE−Req). When a node receives such a packet, it checks if it has already processed the same request; in this case it drops the packet in order to avoid flooding the same request more than once; otherwise, it adds its own ID to the header of the packet and retransmits it. In this way, a list of valid hops is automatically created on the way toward the destination. The process continues until the packet reaches a node with hop count 1 (an end node), that can reach the sink directly. This node changes the option field in the Path Establishment packet to Answer (PE−Ans), and sends it back to the source via the same hop list contained in its header. Additionally, upon forwarding the answer, every relay node reads the header of the packet and records the valid path in its own routing table.

If a (bi-directional) path to the sink exists, the source will eventually receive one or more PE−Ans packets. The node continues to listen to the channel for these answers for a given time, and chooses the best options among those received according to the routing metric. Recall that in any event, when an end node receives a Data packet, it forwards it directly to the sink.

D. Discussion on Routing Metrics

When receiving multiple PE−Ans, a node employs a routing metric to sort the goodness of the paths described in each PE−Ans. SUN supports both local and end-to-end metrics, provided that the information required to compute these metrics is included in the header of the PE−Ans packet. The current implementation of SUN defines two metrics: *Lowest Hop Count* and *MaxMin SNR*.

Lowest Hop Count—This leads to the choice of the path with the lowest number of hops. Ties are broken by keeping the most recent path. The metric is very simply computed by counting the hops in the header of the PE−Ans packet.

MaxMin SNR—In this case, the header of PE packets contains a record of the minimum SNR experimented over all links along the path to an end node. This field is initially set to the highest possible value by the source. When a node receives a PE−Req, it measures the receiver-side SNR and updates the field accordingly. At the end, the metric field will contain the lowest SNR in the path between the source and an end node. When the source receives a PE−Ans, it will update its routing table if the minimum SNR recorded in the packet header is greater than the minimum SNR of the currently known path. The use of the MaxMin SNR metric can result in the choice of more reliable paths. However, PE packets contain one additional field, (hence leading to a slightly greater overhead and processing delay to update the minimum SNR); in addition, longer paths are chosen, and a higher number of relays means a higher number of transmissions that may interfere, both along the same path and across different paths.

We remark that any metrics can be implemented, so long as they can be computed locally, or can be contributed to by the nodes as Path Establishment packets travel throughout the network. Examples of relevant metrics currently being considered for implementation are the expected number of transmissions (ETX) and the expected transmission time (ETT).

E. SUN Dynamic Enhancements

SUN's behavior may be tuned to adapt the protocol to different scenarios. We implemented two mechanisms that operate on two internal parameters of the protocol, namely, the time interval between two consecutive buffer lookup times, and the minimum interval between two PE−Req sent. These enhancements allow SUN to infer the status of the network and to exploit it in order to optimize its performance. In Section IV-A, we compare the performance of SUN with and without the mechanisms described in this section, and show that such mechanisms improve SUN's packet loss rate, energy consumption and end-to-end delay.

1) Round-trip time-adaptive buffer lookup: The buffer lookup interval is typically fixed by the user, and used by the protocol also as a reference for ACK timeouts. The optimal value for this interval depends on several aspects, such as the distance among the nodes, the packet processing time, and the network load. Thus, it may be difficult to calculate it a priori. SUN offers a mechanism that adapts this timer dynamically according to the state of the network. Define the round-trip-time (RTT) as the time that elapses from the transmission of a packet to the reception of the corresponding ACK. SUN behaves as follows:

- if the buffer lookup interval is shorter than the RTT, SUN increases the interval value;
- if the observed RTT decreases or increases, it means that either the position of the nodes changed, or the local sound speed profile varied due to environmental changes, and SUN correspondingly decreases or increases the buffer lookup interval in order to adapt it to the new network configuration.

2) Packet loss-adaptive buffer lookup: If the buffer of a node is non-empty most of the time, and yet the packets sent keep being acknowledged, it typically means that the node is a relay where many routes converge, and that the links to its next hops upstream are good. Thus, the node may try

Table I
GLOBAL SIMULATION PARAMETERS

Parameter	Value
Traffic generation time	10000 s
Simulation time	12000 s
Number of iterations	50
Carrier Frequency	26 kHz
Bandwidth	16 kHz
Bit rate	13.9 kbps
Number of nodes	10
Buffer size	1 kByte

to send `Data` packets at a higher rate. This would allow the node to free space in its buffer for new packets, and hence to reduce the chance that newly received or generated ones are discarded. If SUN recognizes that the node is in this state, it reduces the buffer lookup interval. Otherwise, if the node experiences packet losses upon packet relaying in excess of a predetermined threshold, SUN increases the interval.

F. Differences from classical source routing approaches

SUN is based on the dynamic source routing paradigm: routes are dynamically chosen by the source based on incoming answers to `Path Requests`, and the path to be followed is fully described in the header of each `Data` packet. This approach has been chosen because it leads to several well-known advantages: it does not require to periodically probe the state of the routes; it is straightforward to avoid loops in the paths; routes are created on demand, and it is very easy to repair broken routes; implicit information contained in paths required by other nodes can be cached and reused.

As SUN is based on source routing, some of its features necessarily abide to this paradigm. Consider for example a classical source routing protocol, such as DSR [35] (see also the discussion on DSR features in [37]). Some aspects of SUN and DSR are similar, with special regard to the route discovery and maintenance mechanism.

However, an effective implementation of source routing in underwater scenarios requires several significant changes. First of all, end nodes, i.e., nodes that periodically receive `Probe` packets from a sink (Section III-B), take a very important role in SUN. End nodes provide two main advantages: i) they are offloaded the task to answer `PE-Req` packets on behalf of the sink, which improves the response time of the network and decreases the chance that routing control procedures congest the sink; ii) in the presence of topology changes at the sink, e.g., due to mobility or channel variations, end nodes make it possible to quickly react to these changes. Both advantages are key to improving the effectiveness of underwater communications, and require as little as the periodic transmission of sink `Probe` packets.

In order to counter the high chance that rapid channel quality variations compromise the reliability of the routes, SUN does not allow intermediate relays to answer `PE-Req` packets. Rather, only the end nodes can send `PE-Reqs`. This makes it possible to keep paths fresh, avoids the prolonged reuse of old paths, and makes a much better policy in an underwater scenario. For the same reason, in SUN a maximum validity time is set for each entry in the routing table. This forces the nodes to periodically refresh their paths. Of course, when a fresh `PE-Ans` is returned to a node, all relays along the path can read the `PE-Ans` and benefit from the updated route information. With a similar rationale, SUN keeps track of at most one route per known sink. This policy will be extended in the future by storing multiple routes at no extra overhead, with the caveat that they should be node-disjoint in order to provide a sufficient level of diversity, e.g., in the spirit of [29].

Since underwater acoustic channels are typically asymmetric, SUN allows a node to process a packet only if the node's id is written in the header of the packet. Conversely, overheard routing information cannot be stored and exploited.

The above mechanisms match the behavior of source routing to the peculiarities of underwater acoustic channels, and make it possible to achieve good performance in realistic scenarios. We remark that such results are obtained with no assumptions related to a specific network topology structure, to node location awareness, or to the availability of special hardware at the nodes.

IV. SIMULATION RESULTS

This section presents some preliminary simulation results that prove the validity of our design in both static and mobile networks. All simulations have been performed using DESERT Underwater [38], a framework to DEsign, Simulate, Emulate and Realize Test-beds for Underwater network protocols, which is based on the well known ns2/MIRACLE [39] network simulator. The framework makes it possible to accurately model the behavior of a node that obeys the rules of the SUN protocol; among the options provided by DESERT, we choose to simulate acoustic communications using the implementation of the underwater channel models in [40].

The results of the simulations are grouped into three separate sections. In Section IV-A we analyze the performance of SUN in a static scenario under different traffic generation rates and data payload sizes, proving that the dynamic enhancements developed for SUN (see Section III-E) lead to significant performance improvements. For a representative choice of the parameters, we compare SUN to the ICRP protocol [32] in a static scenario in Section IV-B, and extend the comparison to a mobile scenario in Section IV-C.

The parameters used to configure the simulations are reported in Table I. The parameters of the modems have been set according to the specifications of the S2CR WiSE Underwater Acoustic Modems, by EvoLogics GmbH [41] (the same modems used for the field experiments described in

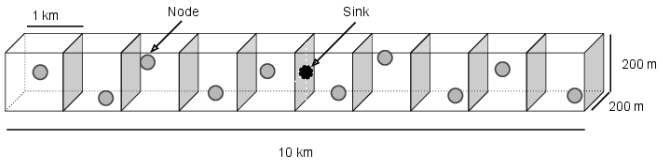


Figure 1. Topology for the static scenario simulations.

Section V). In all runs, the simulation time is set to 12000 s. The `Data` packet generation process continues up to 10000 s, whereas the last 2000 s are left to empty the buffers of the nodes. All packets not delivered to the sink at the end of the simulation are considered lost. The values of the traffic generation rate λ (in packets per node per minute) are chosen to represent practical low and high traffic conditions, from $\lambda = 0.2$, or 1 packet every 5 min per node, up to $\lambda = 6$, or 1 packet every 10 s per node. Scenario-specific parameters are reported in each of the following subsections.

A. SUN with and without dynamic enhancements

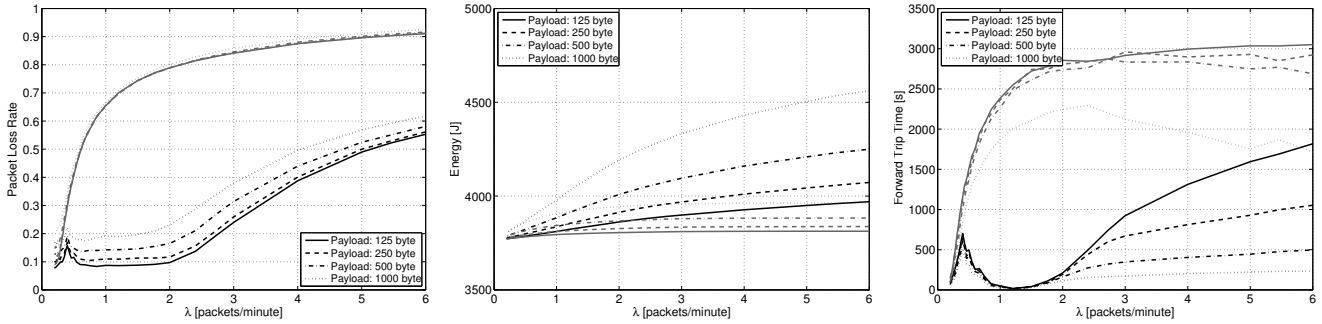
In the static scenario, 10 nodes are positioned in a 3D linear topology. The network area is subdivided into ten regions of size 1000 m $\times$ 200 m $\times$ 200 m. The total network length is therefore equal to 10 km. Each node is positioned at random within a region. A graphical representation of the static scenario is reported in Fig. 1. An additional node that acts as the sink is positioned in the middle of the area at the boundary between the fifth and the sixth region. The power level is set to the minimum that still allows two nodes in two adjacent sections to communicate with each other (in this case, the maximum distance between two consecutive nodes is ≈ 2020 meters, and the source level was set to 138 dB re μPa @ 1 m). With this level, nodes 5 and 6 communicate directly with the sink, and nodes 4 and 7 may also get a direct link, depending on their location. In this analysis, we evaluate three metrics: the end-to-end (E2E) packet loss rate (PLR), defined as the ratio between the number of `Data` packets not delivered to the sink within the simulation time over the total number of packets generated by each node; the energy consumption; and the end-to-end delivery delay, defined as the time that elapses from when a packet is generated to when it is correctly delivered to the sink. The *Lowest Hop Count* metric was chosen for SUN because of the static configuration of the nodes, which makes the path from each node to the sink uniquely determined. (Simulations employing the MaxMin SNR metric were also run and observed to lead to similar results.) The maximum number of retransmissions for each packet was set to 3 in order to achieve a trade-off between the PLR and the E2E delay of correctly delivered packets. The sink probing period was set to 10 min, also because of the static nature of the network.

Each plot in Fig. 2 contains two sets of curves: a gray color denotes the basic version of SUN (which resembles more closely the classical dynamic source routing approach), whereas the black color denotes the version with the enhancements enabled (see Section III-E), where we have set the packet loss threshold for the enhancement in Section III-E2 to 0.2. This makes it possible to see the difference between RTT-adaptive and packet-loss adaptive buffer lookup enhancements. For each set of curves, we report the performance of the protocol for a payload size of 125, 250, 500 and 1000 bytes.

The packet loss rate (PLR), for different payload sizes, is reported in Fig. 2a as a function of λ. We observe that, for $\lambda \leq 0.3$, the PLR is very low, regardless of the packet size and of whether the enhancements are activated or not. Still, the RTT-adaptive buffer lookup times allow the nodes to achieve slightly lower PLR than the basic version of SUN. For higher values of λ, where the packet loss rate would exceed 0.2, the packet loss-adaptive buffer lookup enhancement becomes effective, and helps decrease the PLR substantially. For example, for $\lambda = 1$, the optimized version of SUN achieves a PLR between 0.1 and 0.2, depending on the payload size, whereas the basic version of SUN is affected by a PLR around 0.65. The trend is consistently observed for all values of $0.5 \leq \lambda \leq 2$. For $\lambda \geq 2$ the enhanced version of SUN also suffers from higher traffic load and the increased contention that results, leading to a higher PLR. In any event, the gap between the two versions of SUN remains significant, proving that the enhancements are effective.

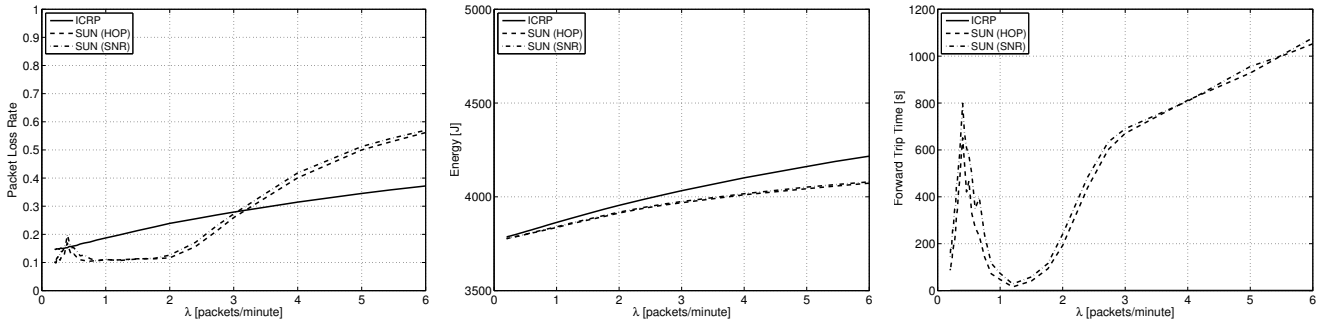
The average energy consumption per node is depicted in Fig. 2b, and is computed as the sum of the energy spent for transmission, reception and idling. The instantaneous power levels required by the modem were set according to the data sheet in [41], where the transmit, receive and idling power are equal to 2.8 W, 1.3 W and 0.285 W, respectively. From Fig. 2b we observe that the amount of energy required by the enhanced version of SUN is almost the same as the basic version for $\lambda \leq 0.5$ (also in this case, the RTT-adaptive buffer lookup enhancement enables slightly better performance with respect to basic SUN). For higher values of λ, the enhanced version of SUN consumes more energy, which is expected due to the higher number of transmissions and receptions that result from its lower PLR. This confirms the better performance achieved thanks to the SUN enhancements. For both versions of the protocol, higher values of λ and higher values of the payload sizes imply higher energy consumption. This is due, respectively, to a higher number of packets to be transmitted and to a larger amount of time spent in transmission and reception rather than idling. We remark that typical Li-ion batteries for commercial modems have a capacity on the order of 10^6 J: in the worst case considered ($\lambda = 6$, 1000-byte payload), a node using the enhanced version of SUN can provide continuous operations for more than 30 days.

The average E2E delivery delay is shown in Fig. 2c. The trends of the curves are the same as in Fig. 2a. We observe that, in general, the enhanced version of SUN behaves better than its basic counterpart, mainly thanks to the dynamic buffer lookup times described in Section III-E1. Such enhancements start being effective for $\lambda \geq 0.5$. Before that, the E2E delay increases up to 700 s, and starts decreasing again when the enhancements limit the persistence of transmission attempts (hence network congestion and likelihood of collisions). Only when the offered traffic becomes overwhelming ($\lambda \geq 2.5$ pkt/min/node) does the E2E delay increase again. In these conditions, the E2E delay increases linearly with λ, and higher payload sizes correspond to a lower average delay. This is because, in congested conditions, one-hop neighbors of the sink get a larger share of network resources for themselves, leading to lower delay for those packets that make it to the sink. This is supported by the similar PLR experienced with different payload sizes, see Fig. 2a.



(a) PLR (b) Energy consumption (c) E2E delivery delay

Figure 2. Performance of SUN with (black lines) and without enhancements (gray lines).



(a) PLR (b) Energy consumption (c) E2E delivery delay

Figure 3. Performance of SUN with enhancements using the lowest hop count metric (dashed lines) and the MaxMin SNR metric (dot-dashed lines), against ICRP (solid lines) in the static scenario of Fig. 1.

B. SUN against ICRP in a static scenario

We proceed by comparing the performance of SUN (with enhancements enabled) against that of the ICRP protocol, which has been introduced in Section II. Because the network is static, the parameters of SUN are very similar to those employed in Section IV-A. In more detail, we set the route validity and probe validity timers to $+\infty$, and the sink probing period to 10 min. We allow up to three retransmissions per packet. The last two parameters are equally set in ICRP. The location of the nodes is the same used in the static scenario (Fig. 1).[3] For SUN, we considered both the Lowest Hop Count and the MaxMin SNR metrics. We will discuss only the case of a 250 bytes payload, but the results can be easily extended to other payload sizes. Fig. 3 reports (a) the PLR, (b) the energy consumption and (c) the E2E delivery delay for both protocols.

From Fig. 3a, we observe that the PLR of SUN is significantly lower than that of ICRP for $\lambda \leq 3$ pkt/min/node. An exception is observed at $\lambda \approx 0.3$, where the SUN enhancements start adapting the protocol behavior to the network load. For $\lambda > 3$ the network becomes saturated (SUN's PLR steadily increases with increasing traffic). In this situation, the flooding approach of ICRP becomes more convenient than SUN's route establishment mechanism, and ultimately makes it possible to deliver a larger number of packets to the sink. In any event, ICRP's PLR also increases steadily with traffic. We observe that the two SUN metrics lead to similar results, as could be expected as a consequence of the empirical propagation model embedded in the DESERT Underwater libraries. However, the result also confirms that both metrics are working well, and can therefore be used in real-life experiments, where the variability of channel conditions over time and space will lead to different results. This will be shown and discussed in more detail in Section V.

The energy consumption plots (Fig. 3b) show similar trends for SUN and ICRP. The higher energy consumption of ICRP is a clear indication of the larger amount of resources required by a flooding-based approach. More significant differences are found by observing the average E2E delay (Fig. 3c). Because the scenario is the same of Section IV-A, the performance of SUN if quite similar to the results presented in that same section. We stress that the MaxMin SNR metric achieves slightly higher E2E delay than Lowest Hop Count metric: in fact, the hop count of each node can vary across different topologies, and the MaxMin SNR metric may prefer a path with more hops. Since ICRP does not have a buffer mechanism at the network level, and since the first packet is sent in broadcast without introducing additional path discovery delays, the only significant sources of delay are the MAC protocol and the propagation time (a few seconds altogether for all values of λ). In addition, with ICRP, most of the packets delivered to the sink come from nodes with a low hop count. Conversely,

[3]We stress that ICRP does not include network-layer retransmissions. Therefore, the mechanism has been delegated to the MAC layer in our simulations.

SUN delivers more packets generated by nodes with a high hop count: therefore, the need to traverse FIFO queues over multiple hops leads to higher E2E delay values.

C. SUN against ICRP in mobile scenarios

In this section we discuss a comparison of SUN (with the enhancements enabled) and ICRP in a mobile scenario where the nodes move in 3D within a volume of 5 km $\times$ 5 km $\times$ 100 m according to a Gauss-Markov (GM) model [42] with self-correlation parameter $\alpha_{gm} = 0.9$, which yields smooth trajectories akin to those that would be traveled by a typical autonomous underwater vehicle.

The parameters of SUN have been configured so as to reflect the network dynamics. In particular, the route validity timer has been set to 600 s, the probe validity timer to 150 s and the sink probing period to 60 s. ICRP's route validity timer has also been set to 600 s. For both protocols, up to three retransmissions are allowed per packet. The results of the comparison between ICRP and SUN (where both the Lowest Hop Count and the MaxMin SNR metrics are used for the latter) are reported in Fig. 4.

The PLR, in the case of the GM 3D mobility model, is reported in Fig. 4a as a function of different generation rate values (λ). Compared to the static scenarios discussed in Sections IV-A and IV-B, in this case the average distance between the source and the destination is smaller throughout the simulation. This leads to a generally lower PLR for both SUN and ICRP than in the static scenario, and to best SUN performance when the *Lowest Hop Count* metric is used. The MaxMin SNR achieves very similar results. The results also suggest that neither SUN nor ICRP suffers substantially from network mobility. On the contrary, SUN's route discovery and renewal mechanism profitably exploits the changing topology of the network, better than ICRP's flooding-based mechanism.

The energy consumption trends (Fig. 4b) confirm that the path establishment mechanism improves the effectiveness of the relaying process, despite the additional overhead introduced by this mechanism. The routing mechanism used by ICRP is not as effective. In fact, ICRP resets routes after a predetermined amount of time (even in the presence of a high route validity time, as set in these simulations), and falls back to flooding after that, leading to a higher energy consumption. In this scenario, where the number of neighbors is typically larger than in the static scenario of Section IV-B, this effect is even amplified.

The E2E delay is shown in Fig. 4c. The conclusions drawn for the static scenario results in Section IV-B are valid here as well: due to the flooding approach of ICRP, whenever a node can deliver a packet, the E2E delay is typically close to one half the round-trip-time. In any event, this mechanism consumes more energy and leads to more frequent collisions, as seen from ICRP's worse PLR. The performance of SUN remains closer to that of ICRP for sufficiently low values of λ mainly due to the RTT-adaptive buffer lookup time in this case, as the packet loss rate does not exceed the preset threshold of 0.2 for low λ.

V. Field Experiments

In this section we present six field experiments that have been conducted to test the adaptive mechanisms of the SUN protocol in a real-world environment. To this end, we deployed six EvoLogics S2CR WiSE Underwater Acoustic Modems [41] in the Werbellin lake, near Berlin, Germany. The typical depth of this environment varies between 15 and 30 m. All experiments were conducted in the month of July. The topologies employed during this experimental campaign are shown in Fig. 5.

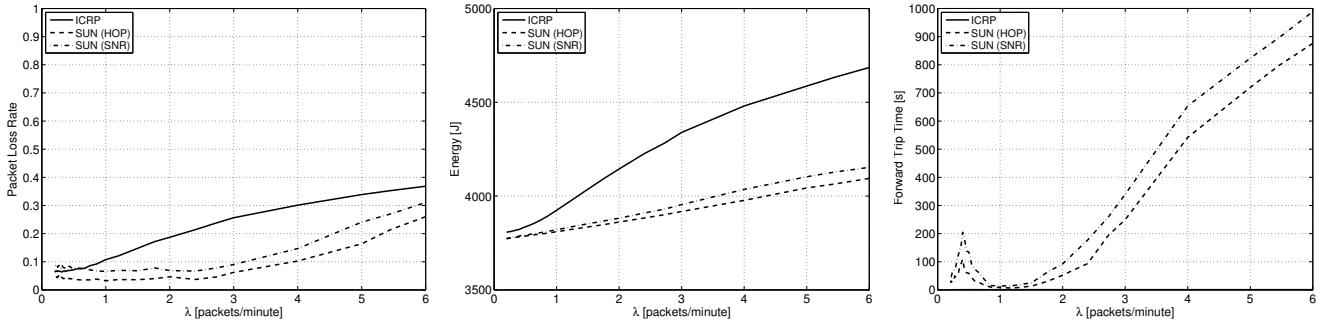
We installed the DESERT Underwater framework [38], [43] on all modems. This provides all libraries required to interface the modems and run, on each of them, a complete network protocol stack. This stack includes: for the application layer, a traffic generator that, for each transmitter, injects packets into the network according to a Poisson process with user-defined rate; for the transport layer, a User Data Protocol (UDP) and, for the network layer, the SUN protocol presented in Section III. For the data link layer, instead, the Medium Access Control (MAC) actually used is D-MAC, implemented as part of the modem firmware [1]. We remark that this experiment serves to measure the performance of the routing protocol. For this reason, we do not implement push-back mechanisms that force the application layer to stop requesting packet transmissions to the routing layer when, e.g., no valid route is known.

Each deployed node could act as a transmitter, relay or sink; by changing the roles of the six nodes of the underwater networks in Fig. 5a, we ran these six experiments:

1) *Relay failure*, which forces SUN to recover valid routes after the failure of a relay node;
2) *Sink failure*, where SUN must recover after the failure of a sink node, and when multiple sinks exist;
3) *Sink detection*, which aims at testing the behavior of SUN when an additional sink node is discovered and considered more convenient by the SUN routing metric;
4) *Static Network with Integrity Metric*, where the route selection is based on the link quality measured by the nodes;
5) *Mobile sink with Integrity Metric*, where SUN is tested in the presence of a mobile sink, and of the route disruption that results;
6) *Mobile sink with Integrity Metric on a larger area*, which aims at stressing SUN in the presence of a mobile sink and with a smaller node density.

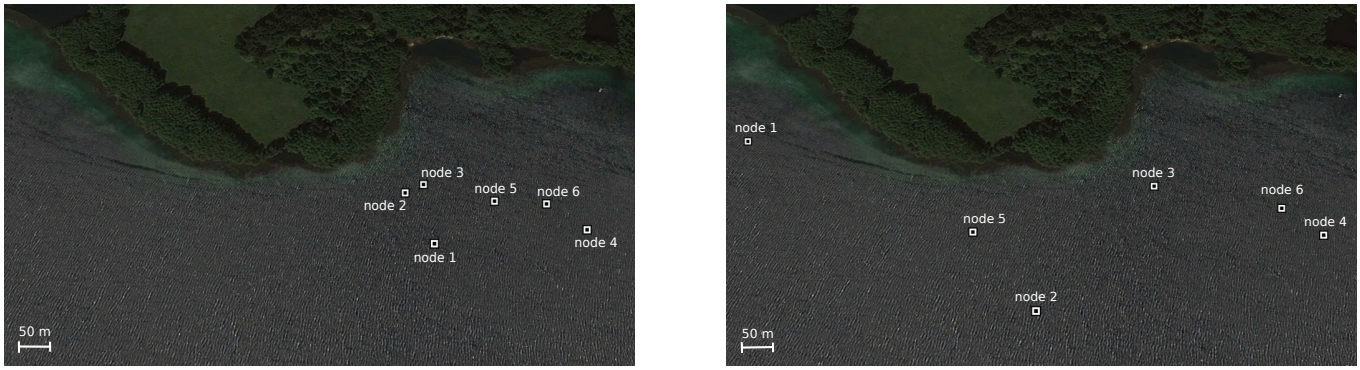
In experiments 1 to 3, SUN used the lowest hop count metric to choose the best route, whereas in experiments 4 to 6 we employed the *integrity metric*, i.e., a value reported by the Evologics modems to measure the quality of the signal;[4] the ARQ mechanism used by SUN (see Section III) has also been enabled, and makes it possible to perform up to one retransmission of a packet for which no acknowledgment has been received. For each experiment, we recorded all packets transmitted, and post-processed the data in order to determine

[4]The higher the integrity metric, the better the quality of the signal. The integrity metric takes into account both the Signal-to-Noise Ratio (SNR) and the physical characteristics of the incoming synchronization preamble.



(a) PLR (b) Energy consumption (c) E2E delivery delay

Figure 4. Performance of SUN with enhancements using the lowest hop count metric (dashed lines) and the MaxMin SNR metric (dot-dashed lines), against ICRP (solid lines) in a mobile scenario with 3D Gauss-Markov mobility.



(a) Network deployment for Exp. 1–5 (b) Network deployment for Exp. 6.

Figure 5. Deployment of S2CR WiSE Underwater Acoustic Modems in the Werbellin lake, Landkreis Barnim, Brandenburg, Germany.

the end-to-end delay, the routes followed by each packet, the lost and dropped packets, as well as the overhead, computed according to the following formula:

$$\big(\sum_i \beta_i C_i\big)/T_B\,, \tag{1}$$

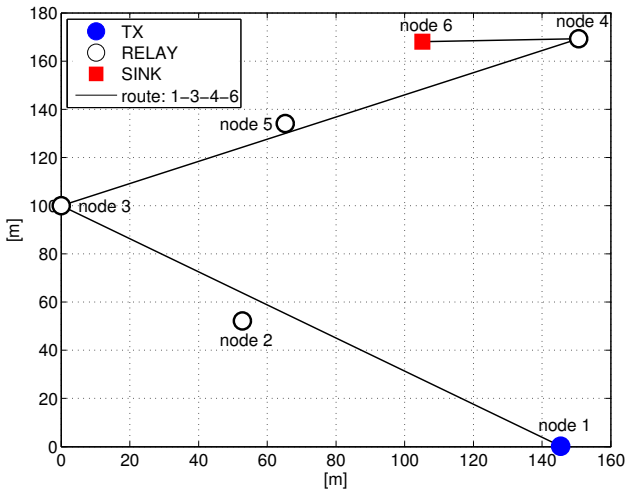
where β_i is the total number of bytes used for each control packet i (`Probe`, `PE–Req`, `PE–Ans`, `ACK` or `PathError` packets), C_i is the number of times each type of control packet has been observed and T_B is the total number of bytes transmitted (both data and control) in the considered network during the experiment.

A. Experiment 1: Relay Failure

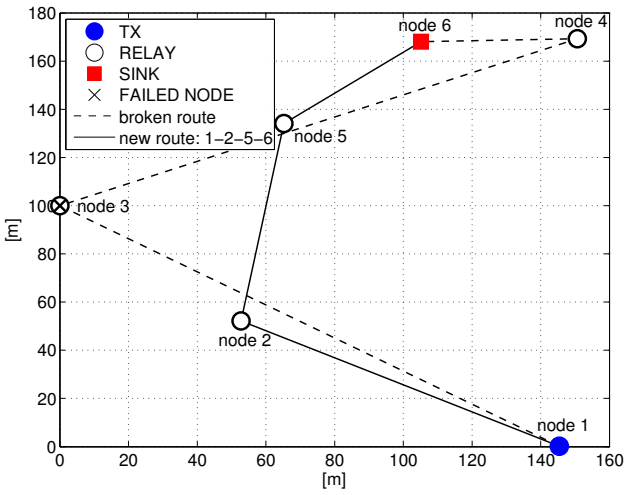
In this experiment, we consider one transmitter (node 1), one sink (node 6) and four relay nodes (nodes 2 to 5). At first, we force SUN to route packets through the path 1–3–4–6, as illustrated in Fig. 6a; practically, to do so we both verified the connectivity throughout the desired path and masked other possible existing connections by dropping packets coming from undesired sources[5] at each node. We note that this mechanism makes it possible to arrange the network in a similar topology as in the simulations discussed in Section IV-A, and therefore to test the adaptive features of SUN in a controlled scenario. More complicated time-varying topologies will be discussed later in this section.

During Exp. 1, we cause the failure of node 3, hence disrupting the route already established by SUN. We make a new route available instead, i.e., route 1–2–5–6, as illustrated in Fig. 6b. This experiment allows us to observe the behavior of SUN in case of relay failure: in Fig. 6c we use circles to indicate the packets that have been correctly sent from the transmitter to the receiver, both before node 3 fails (white-filled circles) and after the recovery of a new route (blue-filled circles); a vertical red-dashed line indicates when the relay failure occurs. Both before and after the relay failure, we observe that the packets are correctly delivered to the sink with a delay between 9 and 40 s, depending on the variable channel conditions and the corresponding number of retransmissions required per link. Furthermore, along the x-axis, we mark the losses of `Data` packets using crosses (i.e., the inability to forward a packet to the next hop along a path because the maximum number of retransmissions has been reached); with diamonds, instead, we mark `Data` packets dropped by SUN because of buffering timeouts. Immediately after the node failure, we can observe a packet loss and three packet drops: the failure of the established path 1–3–4–6 caused the packet loss, while drops are caused by the time required to discover

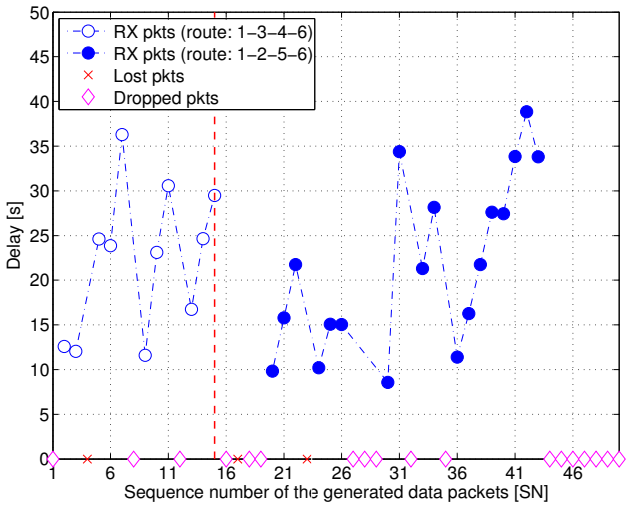
[5]This blacklisting mechanism is implemented within the DESERT software engine commanding the nodes. Once the physical connectivity among the nodes in the network is verified, this solution allows us to have full control on the logical network connectivity during all the experiments performed.



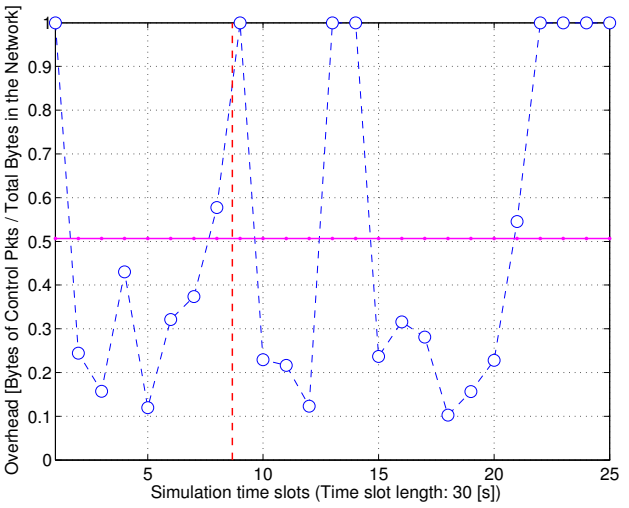
(a) Illustration of the initial route.



(b) Illustration of the node failure and the new route.



(c) Delay for the received packets; lost and dropped packets.



(d) SUN protocol overhead throughout the experiment.

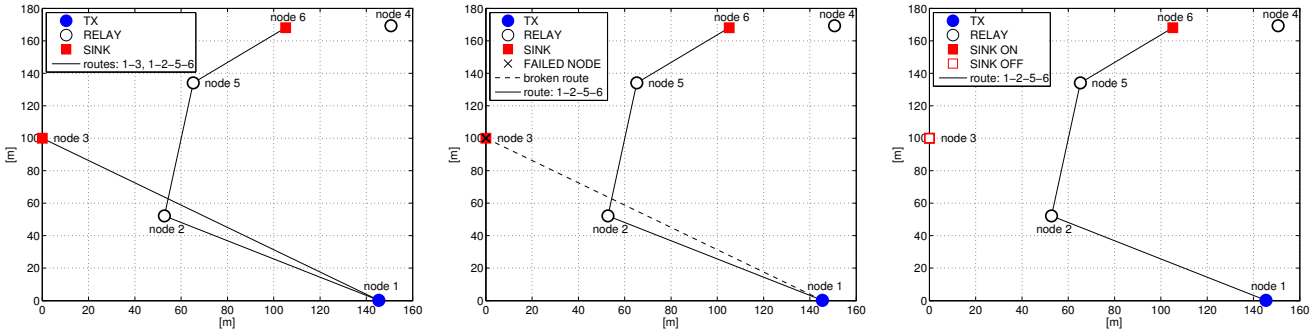
Figure 6. Experiment 1 (Relay Failure). (Panels (a)–(c) are from [1].)

a new route by circulating `PE-Reqs` and `PE-Ans` packets. As expected, packets are dropped mostly when no route is available. This includes i) the beginning of the experiment, before a valid route is found; ii) throughout the experiment, where channel conditions may cause both packet losses and route disruptions which, in turn, lead to dropped packets (we recall that we are manually forcing a single route to be present at any given time in this test); iii) the end of the experiment, where the bad channel conditions do not allow the transmitter to recover a valid route before completely emptying its buffer.

The protocol overhead observed during the experiment is reported in Fig. 6d where we divide the time axis in slots of 30 s and, for each time slot, we compute the ratio between the bytes used for control packets and the total number of bytes transmitted in the network over that time slot according to (1). Since there is only one source, one destination, and one path at any given time in the experiment, the overhead is identically one when only control packets circulate in the network. This occurs when: i) the experiments begin and the routes must be discovered for the first time; ii) a node fails (event marked with a red-dashed vertical line) and a completely new route must be recovered; iii) an error occurs along a path (see the red cross in Fig. 6c corresponding to the loss of the source packet with sequence number 23); and iv) the packet generation process stops, and only the sink node remains active to transmit `Probe` messages. During the rest of the experiment, the overhead is 0.25 or less, since it actually increases only when the SUN protocol has to react to adverse network events.

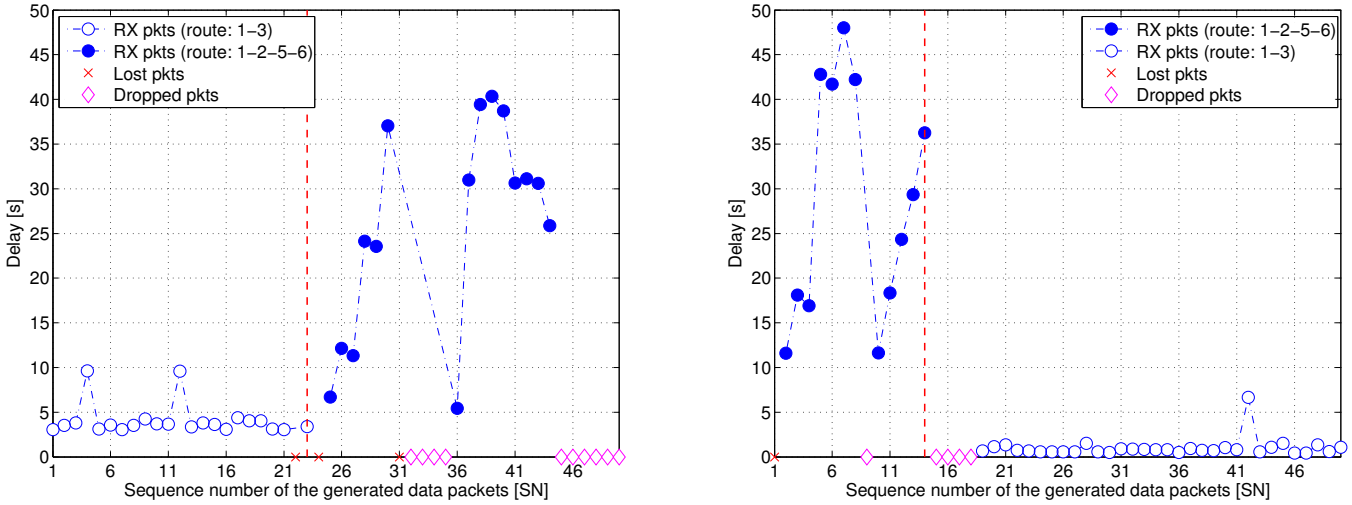
B. Exp. 2 and 3: Sink Failure and Sink Detection

The experiments presented in this section are meant to trigger a reaction by SUN when: 1) the source node has to choose among multiple discovered routes, according to the metric in use (in this case, the lowest hop count metric); 2) multiple sink nodes exist, and they either fail, thereby disrupting a valid route, or appear, thereby enabling the switch to a more convenient route. For these experiments, we consider



(a) Routes used in both Exp. 2 and 3 (b) Node failure in Exp. 2. (c) Initial network scenario for Exp. 3.

Figure 7. Network topologies for Exp. 2 (Sink Failure) and Exp. 3 (Sink Detection). (a) shows the routes initially available in Exp. 2 (before sink 3 fails), which are the same available at the end of Exp. 3 (after sink 3 appears again). (From [1].)



(a) Exp. 2 (Sink Failure) (b) Exp. 3 (Sink Detection)

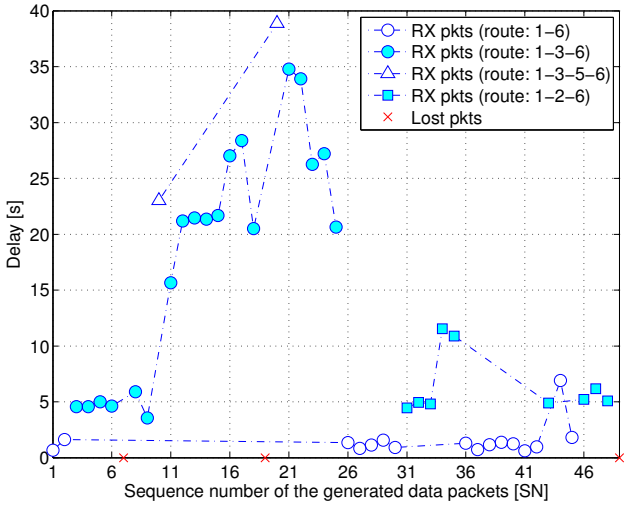
Figure 8. Experiments 2 (Sink Failure) and 3 (Sink Detection). Delay for received packets; lost and dropped packets. (From [1].)

one transmitter (node 1), two sinks (nodes 3 and 6) and three relays (nodes 2, 4 and 5). Fig. 7a illustrates the routes that are enforced via blacklisting in Exp. 2, before a sink fails as illustrated in Fig. 7b. The same routes are available in Exp. 3 when node 3 joins the network: before that, however, only one sink is available, as illustrated in Fig. 7c.

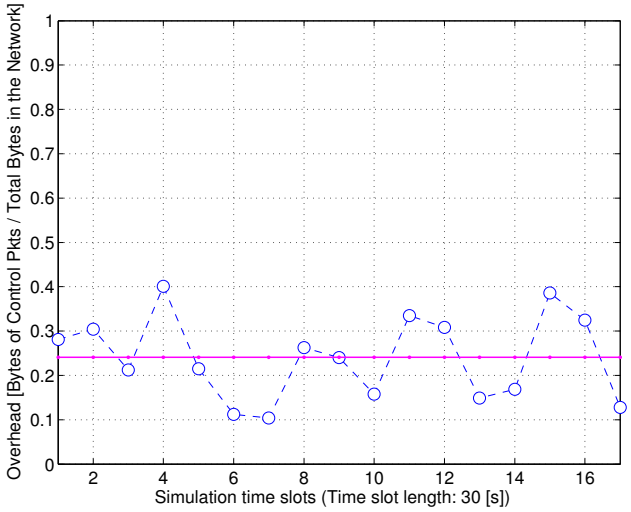
As expected, in both cases, when the two sinks are simultaneously available, SUN picks the most convenient route (i.e., the one with the fewest hops), and forwards `Data` packets directly to node 3 as illustrated in Figs. 8a and 8b (white-filled circles), where the dashed red vertical lines indicate when node 3 fails or joins the network, respectively. From these pictures we see that picking the route with fewest hops causes, in both experiments, a drastic reduction of the `Data` packet delivery time. The PLR measured in Exp. 2 and 3 is 0.26 and 0.12, respectively (0.1 and 0.02 without considering packets dropped in the absence of a valid path, during the route recovery mechanism).

C. Exp. 4: Static Network with Integrity Metric

With this experiment we observe the behavior of SUN when no links are blacklisted, hence the logical topology of the network is not pre-arranged via software. Unlike in experiments 1–3, here we rely on the integrity metric provided by the Evologics modem. We observe that the underwater channel induces quite fast integrity variations even in our simple scenario; in any event, SUN adapts to these changes by choosing the best route among the available paths. Fig. 9a illustrates the delay of the received packets, as well as the occurrence of lost and dropped packets. The different paths chosen by SUN are highlighted with different markers: 34% of the packets received by the sink came via the direct path 1–6; 43% using route 1–3–6; 19% via route 1–2–6 and 4% via the three-hop path 1–3–5–6. Because the path selection is based on the integrity metric, Fig. 9a proves that the highest-integrity path may not be the one with the fewest hops. As a result, the delivery delay changes as a function of the number of hops to be traversed and of the number of retransmissions along each path. Still, the measured PLR achieves a low value of 0.06. Fig. 9b reports the overhead observed during the



(a) Delay for received packets; lost and dropped packets.



(b) SUN protocol overhead throughout the experiment.

Figure 9. Experiment 4 (Static Network with Integrity Metric).

experiment, which was about 0.25 on average throughout the network operation time. This indicates sufficiently stable links, with little need for route rediscovery.

D. Exp. 5 and 6: Mobile Sink with Integrity Metric

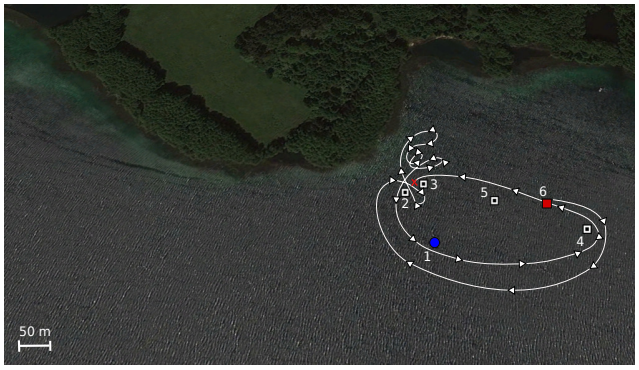
With the last two experiments we observe the behavior of SUN in hybrid networks, where both fixed and mobile nodes are present. As a path selection metric, we employ the integrity metric introduced in the previous subsection. For Exp. 5, we consider the same network topology used for the previous experiments; in addition, by means of a rubber boat, we drag node 6 (the sink) along the path sketched in Fig. 10a. The red-cross indicates the final position of the sink. For Exp. 6, instead, we wish to force multihop routes and stress the SUN protocol. To achieve this, we re-deploy the nodes to create a network covering a larger area, as reported in Fig. 5b, and then move the sink along the path illustrated in Fig. 10b. To introduce further impairments, we also pull the sink out of the water during the experiment (in Fig. 10b this event has been reported with an up-oriented orange arrow) and put it back into the water again (down-oriented green arrow) before the conclusion of the experiment. The average speed of the moving sink, in both Exp. 5 and Exp. 6, is about 5 knots.

Fig. 11 illustrates the packet traffic (classified into `Probe`, `PE-Req`, `PE-Ans`, `Data`, `ACK` and `Path Error` packets) observed during Exps. 5 and 6, by dividing the time axis into slots of 30 s. Figs. 12 and 13 report, instead, the packet delays and the overhead measured during the two experiments. In both cases we can observe that SUN effectively reacts to the dynamic network topology induced by the moving sink. Different routes are chosen throughout the experiments, involving up to 3 hops. In Exp. 6, in particular, the nodes are deployed over a larger area, which generally translates into less reliable links. In this situation, the SUN protocol typically selects longer paths: during Exp. 6 we do not observe any direct transmission between the source and the sink. Instead, transmissions take place through routes with 3 to 4 hops. Fig. 12a shows this fact by means of a different curve for each route. For example, route 1–2–6 (white circles) is used during the first portion of the experiment, then again during the last third. In the intermediate part of the experiment, routes 1–5–3–6 (stars) and 1–3–6 (blue circles) are also employed. We remark that these routes are chosen dynamically, as can be inferred by the fact that the curves intersect and alternate over time.

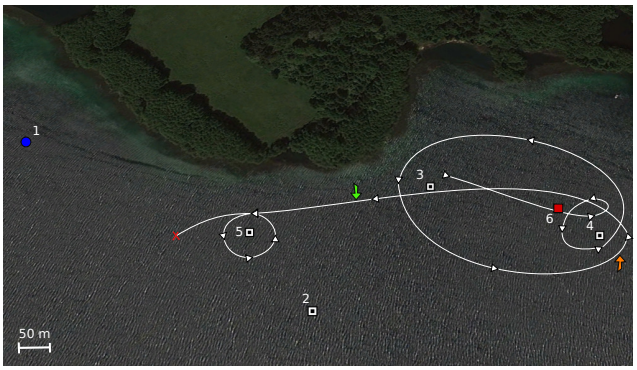
In the middle of Exp. 6, the sink is pulled out of the water, causing many packet losses (see packets with sequence number between 31 and 61 in Fig. 13a) and an increase in the number of control packets exchanged to re-construct a valid route. This can be seen both in Fig. 13b and in Fig. 11b. In particular, the increase of `PathError` messages and corresponding `Path Requests` and answers is apparent from slot 15 until slot 25 in the latter figure, as SUN correctly reacts to the sudden disappearance of the sink. Without considering the losses due to the absence of a sink, in both cases, the SUN protocol showed good overall performance: the PLR measured in Exp. 5 is 0.1 with an average overhead of about 0.25; in the case of Exp. 6, instead, the PLR is 0.5 with an average overhead of about 0.41 (0.3 and 0.25, respectively, without considering the experimental phase during which the sink was not in the water). All the above considerations on experiments 5 and 6 allow us to conclude that SUN is able to correctly handle dynamic network scenarios.

VI. Conclusions

In this paper we presented SUN, a reactive source routing protocol for generic underwater acoustic networks. SUN is inspired to DSR and similar protocols developed for wireless sensor networks, but also includes several features to adapt the behavior of the protocol to the underwater environment and to the dynamics of the network traffic and topology. These features have been shown to improve the performance of SUN considerably with respect to a generic source routing

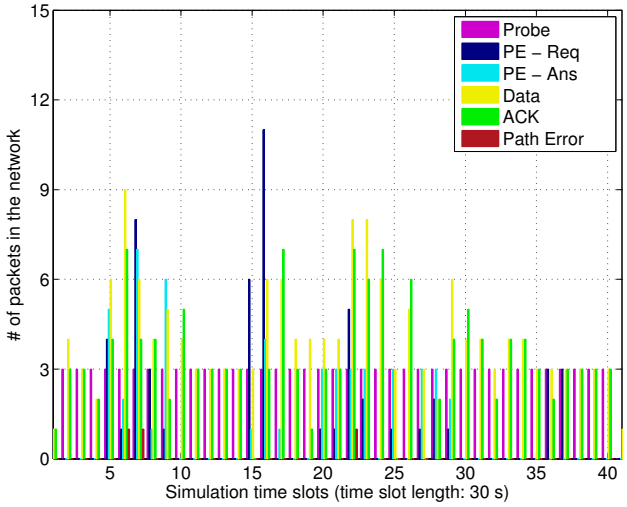


(a) Path followed by the sink in Exp. 5

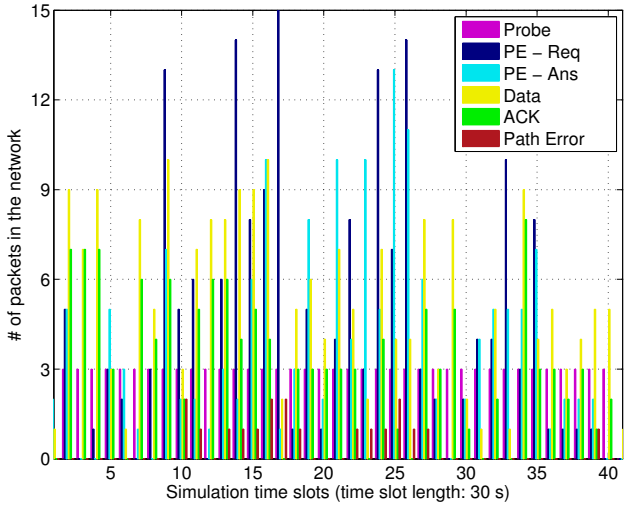


(b) Path followed by the sink in Exp. 6

Figure 10. Hybrid mobile/static network experiments.

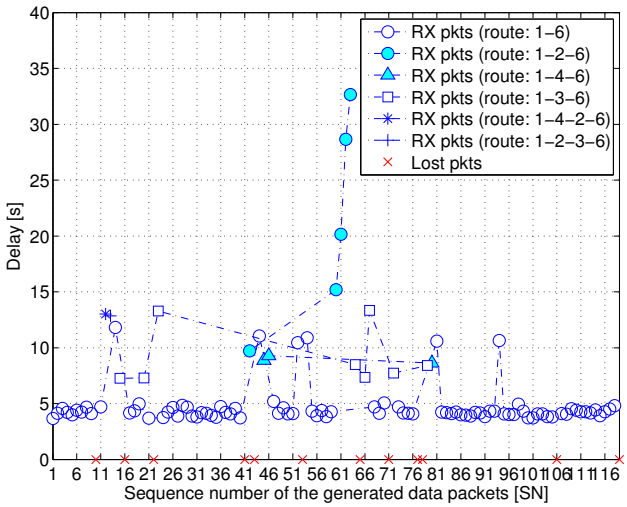


(a) Exp. 5.

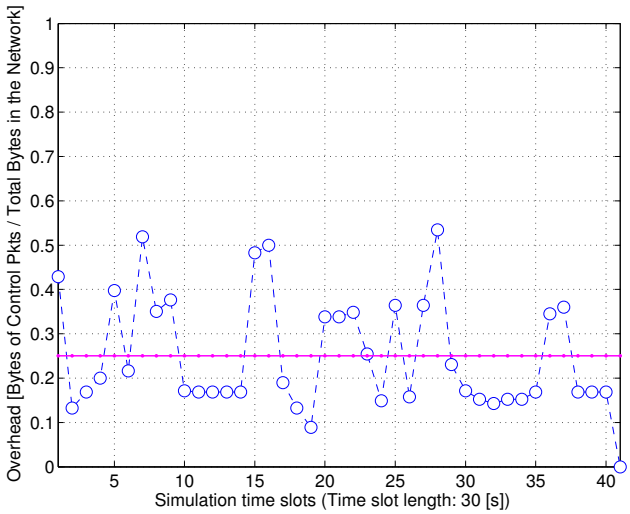


(b) Exp. 6.

Figure 11. Number of control and data packets generated during the Mobile Sink experiments.

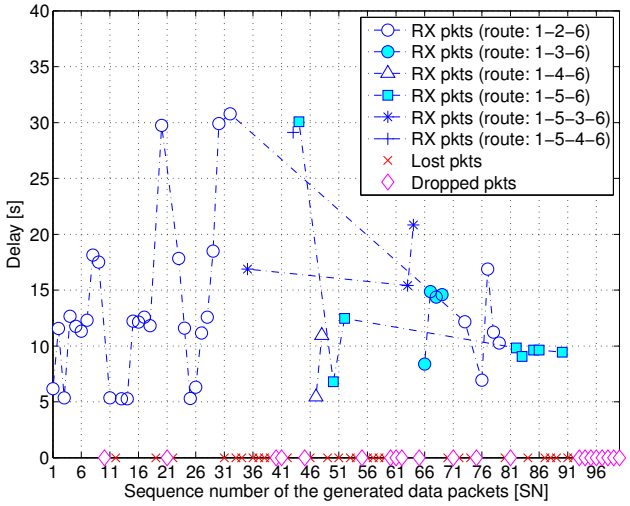


(a) Delay for received packets; lost and dropped packets.

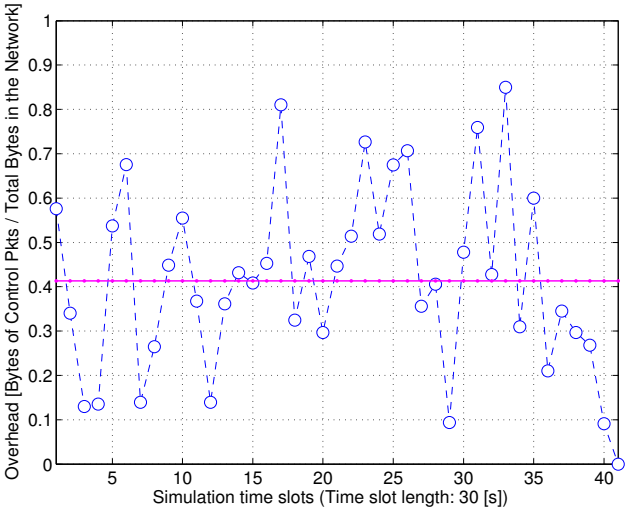


(b) SUN protocol overhead throughout the experiment.

Figure 12. Experiment 5 (Mobile Sink).



(a) Delay for received packets; lost and dropped packets.



(b) SUN protocol overhead throughout the experiment.

Figure 13. Experiment 6 (Mobile Sink – Larger network area).

algorithm. The protocol has been implemented in the DESERT Underwater framework, which was employed to simulate SUN in a variety of environments, and to compare its performance to a competing protocol named ICRP. The results show that SUN achieves lower error rates and energy consumption, while naturally requiring a longer end-to-end delivery delay due to the path discovery and maintenance procedures. The simulations were used as a starting point to design and perform real-world field experiments using both routing metrics provided by SUN. The experiments also confirmed SUN's ability to handle dynamic network topologies involving link quality variations, the insertion and removal of network nodes, and mobile sinks. In addition, we remark that all experiments inherently included mild random movements caused by lake currents. Based on the simulation and experimental results, we conclude that SUN is a feasible solution for both static and mobile underwater acoustic networks.

Acknowledgment

This article is dedicated to the memory of our friend and colleague Giovanni Toso, who tragically died in a car accident on August 10, 2014.

The authors would like to extend their gratitude to the team members of EvoLogics who helped in the various phases of the lake experiment, with special regard to Sergey Yakovlev.

References

[1] G. Toso, R. Masiero, P. Casari, O. Kebkal, M. Komar, and M. Zorzi, "Field experiments for dynamic source routing: S2C EvoLogics modems run the SUN protocol using the DESERT Underwater libraries," in *Proc. MTS/IEEE OCEANS*, Hampton Roads, VA, Sep. 2012.

[2] M. Zorzi, P. Casari, N. Baldo, and A. F. Harris III, "Energy-efficient routing schemes for underwater acoustic networks," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 9, pp. 1754–1766, Dec. 2008.

[3] R. Otnes, A. Asterjadhi, P. Casari, M. Goetz, T. Husøy, I. Nissen, K. Rimstad, P. Van Walree, and M. Zorzi, *Underwater Acoustic Networking Techniques*, ser. SpringerBriefs in Electrical and Computer Engineering. Springer, 2012.

[4] Y. Bayrakdar, N. Meratnia, and A. Kantarci, "A comparative view of routing protocols for underwater wireless sensor networks," in *Proc. IEEE OCEANS*, Santander, Spain, Jun. 2011.

[5] N. Li, J. Martnez, J. M. Meneses Chaus, and M. Eckert, "A survey on underwater acoustic sensor network routing protocols," *MDPI Sensors*, vol. 16, no. 3, Mar. 2016.

[6] R. W. L. Coutinho, A. Boukerche, L. F. M. Vieira, and A. A. F. Loureiro, "Geographic and opportunistic routing for underwater sensor networks," *IEEE Trans. Comput.*, vol. 65, no. 2, pp. 548–561, Feb. 2016.

[7] M. Goetz, S. Azad, P. Casari, I. Nissen, and M. Zorzi, "Jamming-resistant Multi-path Routing for Reliable Intruder Detection in Underwater Networks," in *Proc. ACM WUWNet*, Seattle, WA, Nov. 2011.

[8] C. Tapparello, P. Casari, G. Toso, I. Calabrese, R. Otnes, P. van Walree, M. Goetz, I. Nissen, and M. Zorzi, "Performance evaluation of forwarding protocols for the RACUN network," in *Proc. ACM WUWNet*, Kaohsiung, Taiwan, Nov. 2013.

[9] R. Otnes and S. Haavik, "Duplicate reduction with adaptive backoff for a flooding-based underwater network protocol," in *Proc. MTS/IEEE OCEANS*, Bergen, Norway, Jun. 2013.

[10] R. Otnes, P. A. van Walree, H. Buen, and H. Song, "Underwater acoustic network simulation with lookup tables from physical-layer replay," *IEEE J. Ocean. Eng.*, vol. 40, no. 4, pp. 822–840, Oct. 2015.

[11] N. Chirdchoo, W.-S. Soh, and K. C. Chua, "Sector-Based Routing with Destination Location Prediction for Underwater Mobile Networks," in *Proc. IEEE WAINA workshop*, Bradford, UK, May 2009.

[12] D. Shin, D. Hwang, and D. Kim, "DFR: an efficient directional flooding-based routing protocol in underwater sensor networks," *Wireless Communications and Mobile Computing*, vol. 12, no. 17, pp. 1517–1527, Dec. 2012.

[13] C. Su, X. Liu, and F. Shang, "Vector-Based Low-Delay Forwarding Protocol for Underwater Wireless Sensor Networks," in *Proc. NSFC/IEEE MINES*, Nanjing, China, Nov. 2010.

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

[15] J. M. Jornet, M. Stojanovic, and M. Zorzi, "Focused beam routing protocol for underwater acoustic networks," in *Proc. ACM WUWNet*, San Francisco, CA, Sep. 2008.

[16] H. Yan, Z. J. Shi, and J.-H. Cui, "DBR: depth-based routing for underwater sensor networks," in *Proc. IFIP Networking*, Singapore, 2008.

[17] R. W. L. Coutinho, L. F. M. Vieira, and A. A. F. Loureiro, "DCR: Depth-controlled routing protocol for underwater sensor networks," in *Proc. IEEE ISCC*, Split, Croatia, Jul. 2013.

[18] M. Ayaz and A. Abdullah, "Hop-by-hop dynamic addressing based (H2-DAB) routing protocol for underwater wireless sensor networks," in *Proc. IEEE ICIMT*, San Francisco, CA, Oct. 2009.

[19] S. Basagni, C. Petrioli, R. Petroccia, and D. Spaccini, "Channel-

aware routing for underwater wireless networks," in *Proc. MTS/IEEE OCEANS*, Yeosu, Republic of Korea, May 2012.

[20] Y. Noh, U. Lee, P. Wang, B. S. C. Choi, and M. Gerla, "VAPR: Void-aware pressure routing for underwater sensor networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 5, pp. 895–908, May 2013.

[21] M. Tariq, M. S. Abd Latiff, M. Ayaz, and M. Z. Abbas, "Beacon-based routing protocols for underwater acoustic sensor networks," *Wiley Int. Journal of Commun. Syst.*, Aug. 2017.

[22] R. Diamant, P. Casari, F. Campagnaro, and M. Zorzi, "Routing in multi-modal underwater networks: a throughput-optimal approach," in *Proc. WCNEE (IEEE INFOCOM Workshop)*, Atlanta, GA, May 2017.

[23] R. W. Coutinho, A. Boukerche, L. F. Vieira, and A. A. Loureiro, "Modeling and analysis of opportunistic routing in low duty-cycle underwater sensor networks," in *Proc. ACM MSWiM*, Cancun, Mexico, Nov. 2015.

[24] N. Kanthimathi and Dejey, "Balanced and multi-objective optimized opportunistic routing for underwater sensor networks," *Springer Wireless Personal Communications*, vol. 94, no. 4, pp. 2417–2440, Jun. 2017.

[25] M. A. Rahman, Y. Lee, and I. Koo, "EECOR: An energy-efficient co-operative opportunistic routing protocol for underwater acoustic sensor networks," *IEEE Access*, vol. 5, pp. 14 119–14 132, Jul. 2017.

[26] R. W. L. Coutinho, A. Boukerche, L. F. M. Vieira, and A. A. F. Loureiro, "Design guidelines for opportunistic routing in underwater networks," *IEEE Commun. Mag.*, vol. 54, no. 2, pp. 40–48, Feb. 2016.

[27] W.-G. Seah and H.-X. Tan, "Multipath Virtual Sink Architecture for Underwater Sensor Networks," in *Proc. IEEE OCEANS*, Singapore, 2006.

[28] D. Pompili, T. Melodia, and I. F. Akyildiz, "A resilient routing algorithm for long-term applications in underwater sensor networks," in *Proc. IEEE/IFIP Med-Hoc-Net*, Lipari, Italy, Jun. 2006.

[29] S. Azad, P. Casari, and M. Zorzi, "Multipath routing with limited cross-path interference in underwater networks," vol. 3, no. 5, pp. 465–468, Oct. 2014.

[30] U. Lee, P. Wang, Y. Noh, F. Vieira, M. Gerla, and J.-H. Cui, "Pressure routing for underwater sensor networks," in *Proc. IEEE INFOCOM*, San Diego, CA, Mar. 2010.

[31] M. Domingo and R. Prior, "A distributed clustering scheme for underwater wireless sensor networks," in *Proc. IEEE PIMRC*, Athens, Greece, Sep. 2007.

[32] W. Liang, H. Yu, L. Liu, B. Li, and C. Che, "Information-carrying based routing protocol for underwater acoustic sensor network," in *Proc. IEEE ICMA*, Harbin, China, Aug. 2007.

[33] M. Goetz and I. Nissen, "GUWMANET - multicast routing in underwater acoustic networks," in *Proc. MCC*, Warsaw, Poland, Oct. 2012.

[34] K. Foo, P. Atkins, T. Collins, C. Morley, and J. Davies, "A routing and channel-access approach for an ad hoc underwater acoustic network," in *Proc. MTTS/IEEE OCEANS*, Kobe, Japan, Nov. 2004.

[35] D. B. Johnson, D. A. Maltz, and J. Broch, *Ad Hoc Networking*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001, ch. DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks, pp. 139–172.

[36] E. Carlson, P.-P. Beaujean, and E. An, "Location-aware routing protocol for underwater acoustic networks," in *Proc. MTS/IEEE OCEANS*, Boston, MA, Sep. 2006.

[37] E. A. Carlson, P.-P. J. Beaujean, and E. An, "Location-aware source routing protocol for underwater acoustic networks of AUVs," *Hindawi Journal of Electrical and Computer Eng.*, vol. 2012, Jan. 2012. [Online]. Available: http://dx.doi.org/10.1155/2012/765924

[38] R. Masiero, S. Azad, F. Favaro, M. Petrani, G. Toso, F. Guerra, P. Casari, and M. Zorzi, "DESERT Underwater: an NS-Miracle-based framework to DEsign, Simulate, Emulate and Realize Test-beds for Underwater network protocols," in *Proc. IEEE/OES OCEANS*, Yeosu, Republic of Korea, May 2012.

[39] N. Baldo, M. Miozzo, F. Guerra, M. Rossi, and M. Zorzi, "MIRACLE: The Multi-Interface Cross-Layer Extension of ns2," *EURASIP Journal on Wireless Communications and Networking*, Jan. 2010.

[40] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *ACM Mobile Comput. and Commun. Review*, vol. 11, no. 4, pp. 34–43, Oct. 2007.

[41] "Evologics," Last time accessed: July 2014. [Online]. Available: http://www.evologics.de/

[42] B. Liang and Z. J. Haas, "Predictive distance-based mobility management for PCS networks," in *Proc. IEEE INFOCOM*, New York, USA, Mar. 1999.

[43] P. Casari, C. Tapparello, F. Guerra, F. Favaro, I. Calabrese, G. Toso, S. Azad, R. Masiero, and M. Zorzi, "Open-source suites for the underwater networking community: WOSS and DESERT Underwater," *IEEE Network, special issue on "Open Source for Networking: Development and Experimentation"*, vol. 28, no. 5, pp. 38–46, Sep. 2014.