DOTS: A Propagation Delay-Aware Opportunistic MAC Protocol for Mobile Underwater Networks

Youngtae Noh, *Member, IEEE*, Uichin Lee, *Member, IEEE*, Seongwon Han, *Student Member, IEEE*, Paul Wang, *Member, IEEE*, Dustin Torres, *Member, IEEE*, Jinwhan Kim, *Member, IEEE*, and Mario Gerla, *Fellow, IEEE*

Abstract—Mobile underwater networks with acoustic communications are confronted with several unique challenges such as long propagation delays, high transmission power consumption, and node mobility. In particular, slow signal propagation permits multiple packets to concurrently travel in the underwater channel, which must be exploited to improve the overall throughput. To this end, we propose the delay-aware opportunistic transmission scheduling (DOTS) protocol that uses passively obtained local information (i.e., neighboring nodes' propagation delay map and their expected transmission schedules) to increase the chances of concurrent transmissions while reducing the likelihood of collisions. Our extensive simulation results document that DOTS outperforms existing solutions and provides fair medium access even with node mobility.

Index Terms—Underwater, AUV, medium access control, opportunistic transmission, CSMA

1 Introduction

Mobile underwater sensor networks have recently been proposed as a way to explore and observe the ocean with wide area coverage at reasonable cost when compared to traditional tethered approaches (e.g., seabed sensors) [1], [2], [3], [4]. Towards this goal, a swarm of mobile sensors, e.g., autonomous underwater vehicles (AUVs) such as REMUS and IVER2 or floats such as UCSD Drogues [5], can be deployed to the venue of interest for short-term *ad hoc* real-time aquatic missions such as oil and chemical spill monitoring, submarine detection, and surveillance [6], [7]. Mobile node monitor local underwater activities and report collected sensor data using acoustic multi-hop routing to other mobile nodes for collaboration or simply to a distant data collection center.

Despite the technological advances of acoustic communications, we are still confronted with limitations that need to be addressed in order for UW-ASNs to be put into practical use, namely severely limited bandwidth, long propagation delay (1.5 km/s, five orders of magnitude slower than radio signals), and relatively high transmission power (e.g., more than 100-fold more power consumption than reception [8], [9]).). Moreover, the unreliable nature of underwater wireless channels due to complex multipath fading and surface scattering further aggravates data communications [10].

Under these circumstances, medium access control (MAC) protocols designed for terrestrial packet radio networks cannot be directly used because the propagation delay of acoustic signals is much greater than the packet transmission time (e.g., 0.5 sec versus 0.04 sec to transmit a 256 byte data packet with the data rate of 50 kbps over a 750 m range)—carrier sensing in carrier sense multiple access (CSMA) may not prevent packet collisions. This unique situation, however, permits multiple packets to concurrently propagate in an underwater channel, which must be exploited in order to improve the channel throughput. While this phenomenon is also observed in transatlantic wire lines or wireless satellite links, the main departure is that these are point-to-point links without any contention and that the large bandwidth-delay product is exploited at a higher layer, namely TCP. In general, long propagation latency in an underwater wireless network creates a unique opportunity for *temporal reuse* that allows for multiple concurrent packets propagating within the same contention domain. Note that temporal reuse is an additional opportunity on top of well-known *spatial reuse* in wireless networks which allows concurrent, non-colliding transmissions to different destinations if they are sufficiently removed from one another, solving the exposed terminal problem.

Recently a great deal of attention has been focused on exploiting temporal and/or spatial reuse of acoustic channels to improve the throughput. For instance, slotted FAMA (S-FAMA) uses time slotting in order to lower the

- *Y. Noh is with the Cisco Systems Inc., Milpitas, CA, 95134. E-mail: ynoh@cisco.com.*
- *U. Lee is with the Department of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology, Daejeon, Korea. E-mail: uclee@kaist.edu.*
- *P. Wang is with the Morgan Stanley & Co. LLC. E-mail: Paul.Wang@morganstanley.com.*
- *D. Torres is with the Intel Corporation, Santa Clara. E-mail: dustin.torres@intel.com.*
- *J. Kim is with the Division of Ocean Systems Engineering, Korea Advanced Institute of Science and Technology, Korea. E-mail: jinwhan@kaist.ac.kr.*
- *S. Han and M. Gerla are with the Department of Computer Science, University of California, Los Angeles, CA 90095. E-mail: {swhan, gerla}@cs.ucla.edu.*

Manuscript received 5 Nov. 2012; revised 30 May 2013; accepted 23 Dec. 2013; date of publication 8 Jan. 2014; date of current version 3 Mar. 2014.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TMC.2013.2297703

probability of collisions by aligning packet transmissions into slots (as in slotted Aloha) while Propagation-delay-tolerant collision avoidance protocol (PCAP) [11] allows a node to send multiple reservation requests for transmission time slots (i.e., request to transmit, RTS). In Underwater-FLASHR (UW-FLASHR) [12], time slots are divided into reservation and data transmission periods to realize efficient channel reservation and to minimize data packet losses caused by control packet exchanges. For better channel utilization, most protocols attempt to build a time division multiple access (TDMA) schedule using brute-force learning via repeated trial-and-errors [12] or solving computationally hard optimal scheduling problems as in ST-MAC [13] and STUMP [14]. Distributed approximation algorithms for optimal scheduling were proposed in the literature [13], [14]. However, discovering a reasonable TDMA schedule requires a network-wide consensus, incurring a large number of packet exchanges and taking a considerable amount of time. In general, TDMA-based methods are not suitable for resource constrained underwater mobile sensor networks, because nodes must periodically perform expensive scheduling operations.

Nonetheless the key insights from TDMA-based scheduling methods allow us to enhance conventional CSMA-like random channel access protocols as follows. We need to ensure that transmissions are scheduled carefully such that they do not interfere with the reception of each other's packets by their intended receivers. To satisfy this requirement, each node must evaluate the collision conditions for neighboring packet receptions prior to transmitting a packet. Recall that a *collision* occurs when a receiver tries to decode a packet when more than one packet arrives from different senders simultaneously [15]. The key intuition is that each node can *predict* whether its upcoming packet transmission will collide with another's if it has the neighboring nodes' propagation delay information and their transmission schedules.

In this paper, we consider this idea and propose the delay-aware opportunistic transmission scheduling (DOTS) algorithm designed for underwater mobile sensor networks. The following are the key contributions of the paper.

- One of the key assumptions of DOTS is clock synchronization, because nodes build local propagation delay maps by overhearing packets. Syed et al. proposed a protocol called time synchronization for high latency (TSHL) and validated that TSHL can correct clock offset and skew in a reliable and efficient manner using simulations [16]. In this paper, we implement this protocol on the UANT platform that is composed of a software defined radio and a mix of custom and commercially available hardware for the acoustic transmitter and receiver [17]. We demonstrate that TSHL can effectively synchronize clock offset and skew. To the best of our knowledge, this is the first real implementation of its kind.
- DOTS can effectively exploit temporal and spatial reuse by using local information. In DOTS, each node learns neighboring nodes' propagation delay information and their expected transmission schedules by passively overhearing packet transmissions. Thus, DOTS can compensate for the long propagation latencies by increasing the chances of concurrent transmissions while reducing the likelihood of collisions. Our evaluation results confirm that DOTS can significantly improve the overall throughput. We show that such opportunistic scheduling can effectively handle spatial-unfairness caused by physical location and propagation latency (i.e., the closer the distance between a pair of nodes, the higher the chance of capturing the channel [18]). Further, we validate DOTS's robustness to mobility using a 3D version of the meandering current mobility (MCM) Model [19].
- We propose a simple performance enhancement mechanism that permits multiple outstanding packets at the sender side (multiple transmission sessions). We provide preliminary simulation results of these DOTS variants in representative topologies and show that enabling multiple transmission sessions significantly improves the overall throughput.

This paper significantly enhances our earlier work [20] as follows:

- We provide a review of mobile underwater networks and resource constraints (Section 2) and a thorough review of underwater MAC protocols (Section 2.2).
- We deliver a more complete set of simulation results by considering the meandering current mobility model under various system parameter configurations (Section 5).
- We propose an enhancement technique of enabling multiple transmission sessions which significantly improves the overall performance as opposed to the original DOTS protocol (Section 6).

The rest of the paper is organized as follows. In Section 2, we review the resource constraints of underwater mobile sensors and thoroughly examine underwater MAC protocols. In Section 3, we present our experimental results to demonstrate the feasibility of underwater time synchronization, which is the prerequisite of DOTS. In Section 4, we illustrate the key components of DOTS, namely delay map management, transmission scheduling, and guard time. In Section 5, we conduct extensive simulations to validate the performance of DOTS against that of other well-known underwater MAC protocols. In Section 6), we propose a simple enhancement mechanism that enables multiple transmission sessions and evaluate its performances using representative topologies. Finally, we conclude the paper in Section 7.

2 Background and Related Work

We review the mobile underwater networks, types of mobile sensors, their constraints (e.g., communication characteristics and energy consumption) and then thoroughly examine underwater MAC protocols.

2.1 Mobile Underwater Networks and Resource Constraints

The design of oceanic networks for monitoring and scientific exploratory purposes can be largely classified into two categories: (1) static sensors tethered at the seabed or buoys on the ocean surface with external power sources (e.g., NEPTUNE [21]), and (2) mobile sensors such as AUVs and underwater floats (e.g., SeaWeb [22], ARGO [23], UCSD Drogues [5]). Static sensors are typically used for long term, pre-planned missions such as seismic activity monitoring, whereas battery-powered mobile sensors are used for short-term missions such as oil and chemical spill monitoring. The key benefit of mobile sensing is that mobility permits more flexible underwater exploration with wide area coverage at reasonable cost. AUVs can follow planned trajectories such as a sequence of tracklines, waypoints, and depth excursions [24], while floats have restricted mobility as they move along with water current (e.g., ARGO [23], UCSD Drogues [5]). Given that the cost effective coverage is one of the primary concerns of mobile sensors, such networks must employ low-cost, energy-efficient mobile nodes, and thus, resource constraints must be carefully examined [1].

Mobile Sensor Types. The most common AUV configuration is a torpedo-like vehicle (e.g., REMUS, IVER2) with a streamlined body with propeller and control surfaces at the stern [24]. The speed of such AUVs in the range of 0.5 to 5 m/s, and most vehicles operate at around 1.5 m/s. Another configuration is a glider (e.g., Seagliders [1]) that uses small changes in its buoyancy in conjunction with wings to make up-and-down, sawtooth-like movements. Although gliders have such restricted mobility patterns, due to the energy efficiency, they can provide data collection on temporal and spatial scales that would be costly if traditional shipboard methods are used. Unlike AUVs, underwater floats like UCSD Drouges and ARGO [23] mainly use a buoyancy controller for depth adjustment and passively move along with the water current.

Resource Constraints of Mobile Sensors. We review the resource constraints of mobile underwater sensors, namely *acoustic communications* and *energy consumption*. Communications in the underwater acoustic channel are with two innate characteristics: low bandwidth and large propagation delay. The available bandwidth of the acoustic channel is limited and strongly depends on both range and frequency. As surveyed by Kilfoyle et. al. [25], existing systems have highly variable link capacity, and the attainable range and rate product can hardly exceed 40 km-kbps. The signal propagation speed in the acoustic channel is 1.5×10^3 m/sec, which is five orders of magnitude lower than radio propagation speed 3×10^8 m/sec in the air. This huge propagation delay has great impact on network protocol design. Also, it is important to note that underwater acoustic modems consume significant amount of energy when compared with terrestrial radios; e.g., WHOI Micromodem-1 has the active/receive state with power consumption of 158 mW and the transmission state with full power consumption up to 48 W [9].

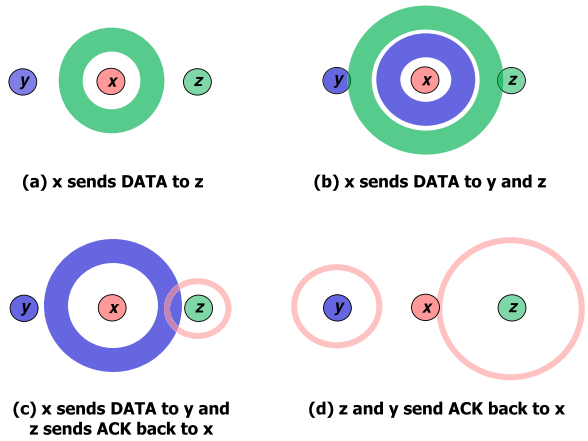


Fig. 1. Temporal reuse.

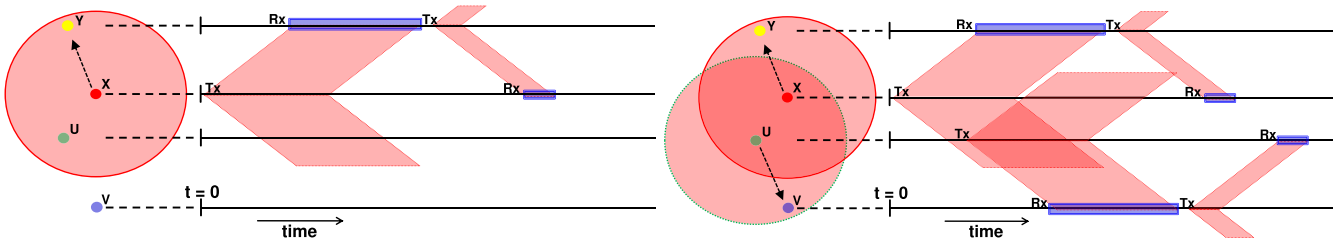
2.2 Review of Underwater MAC Protocols

In multi-hop wireless networks, it is important to efficiently utilize limited network resources and to provide fair access for competing data flows. It has been proven that CSMA provides reasonable performance and fairness [26]. Since CSMA does not require strict scheduling, it can support node mobility, which is also a major challenge in mobile underwater networks. However, the handshaking mechanism of CSMA leads to a severely degraded system throughput due to the presence of long propagation delay of acoustic signals in mobile underwater networks, which is a well-known problem. Moreover, carrier sensing may fail to detect an ongoing transmission due to the propagation delay, which impairs the performance of CSMA protocols [27].

2.2.1 Temporal Reuse

One potential solution for improving CSMA in mobile underwater networks is to utilize *temporal reuse* that exploits the long propagation latencies of acoustic waves. Fig. 1 demonstrates the notion of temporal reuse. Node x sends a DATA packet to node z in Fig. 1a and again at a later time another DATA packet to node y in Fig. 1b. Node z sends an acknowledgment (ACK) back to node x as node y is about to receive the transmission from node x in Fig. 1c. Finally, node y sends an ACK back to node x in Fig. 1d. This case enables the data and ACKs to be transmitted and received without any collision.

To harness this temporal reuse, Yackoski et al. [12] proposed UW-FLASHR, a variant TDMA protocol that can achieve higher channel utilization than the maximum utilization possible in existing TDMA protocols. Hsu et al. [13] proposed ST-MAC, another underwater TDMA protocol that operates by constructing spatial-temporal conflict graph (ST-CG) to describe the conflict delays among transmission links and reduces the ST-CS model to a new vertex coloring problem. A heuristic, called the Traffic-based one-step trial approach (TOTA), is then proposed to solve the coloring problem. Kredo et al. [14] proposed a TDMA-like protocol called STUMP that uses propagation delay information and prioritizes conflicting packet transmissions based on certain metrics (e.g., random ordering and uplink delay ordering). However, TDMA scheduling is typically



(a) Exclusive medium access: *exposed terminal*

(b) Concurrent transmissions w/o causing collision

Fig. 2. Spatial reuse.

performed in a centralized way which is not resilient to failure; moreover, discovering a reasonable TDMA schedule using distributed algorithms for optimized transmission scheduling requires a network-wide consensus. Thus, TDMA-like protocols are not suitable for resource constrained mobile sensor networks.

CSMA-like protocols (or reservation-based protocols) have been proposed to exploit temporal reuse in several ways. Given that channel reservation takes long time (i.e., RTS/CTS), Guo et al. proposed adaptive propagation-delay-tolerant collision avoidance protocol (APCAP) that allows a node to transmit packets in out-of-order during this period (i.e., multiple reservations concurrently) [11], but it does not detail scheduling strategies for out-of-order packet delivery. To reduce the control overhead (e.g., reservation, acknowledgement), R-MAC [28] delivers a burst of packets (or a packet train) and delayed ACKs, thereby improving the channel throughput. Chen et al. proposed ordered CSMA that transmits each data packet in a fixed order [29]. Given the fact that two sequential carriers traveling in the same direction will not collide, each station transmits immediately after receiving a data frame from the previous station sequentially, instead of waiting for a period of maximum propagation delay. Yet, ordered CSMA is not appropriate for large-scale multi-hop networks because generating collision free transmission order requires relative positions of all nodes in the network and a large number of packet exchanges. Ng et al. [30] proposed MACA-U which is a redesigned MACA [31]; i.e., the five-state transition rule by considering long propagation delay, the packet forwarding strategy based on priority, and a binary exponential back-off algorithm. Yet MACA-U only considers the case of handling two neighboring source nodes concurrently transmitting RTS packets, and thus, it does not fully exploit the temporal reuse.

Chirdchoo et al. [32] proposed a receiver initiated reservation protocol called receiver-initiated packet train (RIPT) where after initiating packet transfers, the receiver accepts the packet transmission requests from its neighboring nodes and builds a transmission schedule for its neighboring nodes by considering the propagation delay to its neighbors. In RIPT, the receivers need to periodically initiate packet transfers, which is very expensive, and under varying traffic demands, it is non-trivial to determine when to initiate packet transmissions. Chirdchoo et al. [33] proposed another reservation based protocol, MACA-MN, to improve the channel utilization by enabling multiple packet trains to neighbors. MACA-MN allows a node to send a packet train to multiple neighbors by transmitting RTS with some additional information, e.g., the number of DATA packets for multiple intended neighbors, as well as the inter-node propagation delay from the sender to its intended receivers. Kredo et al. [34] recently examined a range of TDMA-based channel scheduling methods (e.g., node/group/link/slot levels) to determine the best balance between performance and coordination overheads in underwater networks.

Ng et al. [35] proposed reverse opportunistic packet appending (ROPA), a sender-initiated handshaking based protocol where a sender solicits its one-hop neighbors to opportunistically append their packets to the original outstanding packet (packet trains) to increase channel utilization. Further, Ng et al. [36] presented bidirectional-concurrent MAC (BiC-MAC) which further enhances channel utilization of ROPA; a sender-receiver node pair can exchange multiple rounds of bidirectional packet transmissions for every handshake. Unlike existing underwater CSMA solutions, DOTS neither requires an additional phase for reservation scheduling nor restricts transmission schedules to a specific order. DOTS is a sender initiated protocol that relies solely on passively overhearing neighboring transmissions to make intelligent local decisions based upon its own transmission schedule that does not interfere with neighboring receptions, thereby aggressively exploiting temporal reuse.

2.2.2 Spatial Reuse

Spatial reuse in mobile underwater networks also improves the channel utilization by allowing concurrent transmissions. In Fig. 2a, a network topology consisting of four nodes is depicted and its corresponding signal propagation in time is drawn on the side. Node x gains the exclusive access of the channel in its collision domain, preventing node u from transmitting to node v, since node u's transmission will interfere with node x's reception of an ACK from node y, known as the *exposed terminal problem*. However, Fig. 2b shows that it is still possible for node u to transmit concurrently without affecting x's transmission, enabling spatial reuse of the medium. While spatial reuse is well-investigated in terrestrial wireless communications [37], [38], [39], [40], [41], to the best of our knowledge, a little work has been done in mobile underwater networks.

To reduce a TDMA protocol's high dependency on topology information, Diamant et al. [42] proposed a spatial-reuse TDMA scheduling protocol with a broadcast scheduling algorithm called as robust broadcast scheduling problem (R-BSP). R-BSP adapts a combination of an underlying skeleton schedule (obtained from a topology-transparent

schedule) and a topology-dependent schedule, which ultimately provides additional spatial reuse in case of reliable topology information. Ma et al. [43] proposed an efficient scheduling algorithm with constant approximation ratios to the optimum solutions for both unified and weighted traffic load scenarios. This work identifies the spatio-temporal link scheduling problem in UW-ASNs, which is significantly different from terrestrial wireless networks by a new conflict graph (more accurate than slotted spatio-temporal conflict graph in [13]) and provides interference-aware spatio-temporal link scheduling algorithms. Since these protocols are TDMA based protocols, discovering a reasonable TDMA schedule using distributed algorithms for optimized transmission scheduling requires a network-wide consensus. TDMA-like protocols are not suitable for resource constrained mobile sensor networks. Diamant et al. [44] proposed a distributed collision avoidance handshake-based scheduling protocol that makes use of joint temporal and spatial reuse and will be referred to as the joint time and spatial reuse handshake protocol. This protocol improved existing solutions by considering spatial-temporal reuse but their applications are limited to stationary networks. Henceforth, we follow with a short discussion of related works in terrestrial networks and its applicability to mobile underwater CSMA protocols.

MACA-P [45] detects an expose terminal from *Request-To-Send/Clear-To-Send* (RTS/CTS) exchanges such that a node overhears an RTS without overhearing the corresponding CTS. MACA-P introduces a control gap (or delay) between RTS/CTS and DATA/ACK to allow neighboring nodes to schedule their transmissions (via explicit RTS/CTS). Given that control gap incurs an extra overhead, Shukla et al. proposed to use direct data transmissions without RTS/CTS during the exposed period [40]. Alternative method to this approach is to build local conflict maps by empirically detecting expose link pairs via off-line methods (e.g., broadcast collision based interference estimation as in RTSS/CTSS [38]), or on-line methods (e.g., unicast collision based interference estimation as in CMAP [46]). In RTSS/CTSS, nodes coordinate simultaneous transmissions using new control messages, namely the request-to-send-simultaneously (RTSS) and the clear-to-send-simultaneously (CTSS), whereas in CMAP, nodes monitor neighboring nodes' transmissions to opportunistically schedule simultaneous transmissions.

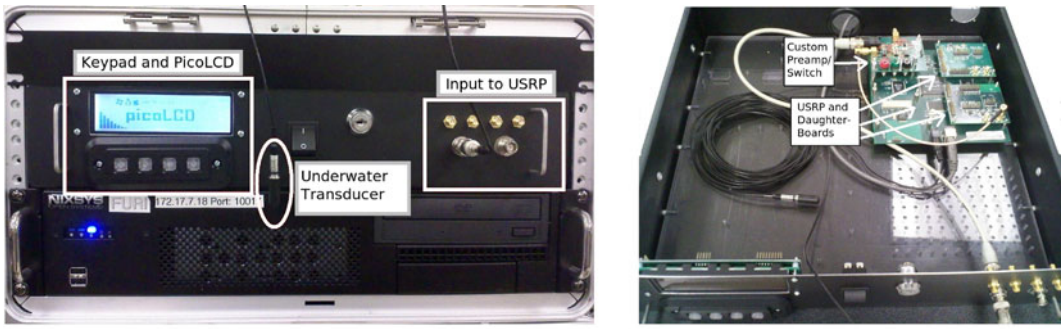
In mobile underwater networks, we note that building such conflict maps is very expensive (due to power-hungry packet transmissions and mobility of sensor nodes), and moreover, they fail to take the large propagation delay into account. In this paper, we propose to use delay maps which can be built by passively observing transmissions. Given such delay maps can be used to predict potential collisions, our approach utilizes delay maps to opportunistically schedule simultaneous transmissions (which can be done without extra control packet exchanges). Note that since we only use delay maps, our approach cannot exploit the capture effect where a receiver can correctly decode a packet even in the presence of other concurrent transmissions. Yet, our approach can be extended to exploit the capture effect as in interference aware (IA) MAC [40], which is the part of our future work.

3 DOTS Prerequisite

It has been shown that observed information obtained from passively overhearing neighboring transmissions can be useful in estimating collisions at the intended receivers [47]. DOTS uses the passively obtained information by building a *delay map* to achieve both temporal and spatial reuse by making intelligent transmission scheduling decisions. DOTS therefore is able to compensate for the long propagation latencies and severely limited bandwidth of the acoustic medium by using passively observed information to increase the chances of concurrent transmissions while reducing the likelihood of collisions. However, the lack of clock synchronization could make it difficult for an overhearing node of a transmission to gauge the propagation delay between itself and the transmitting node. Thus, the DOTS protocol makes the assumption of time synchronization amongst all nodes in the network, similar to existing underwater CSMA solutions proposed in [18], [48], [49]. This assumption is necessary in order to accurately enable estimation of the transmission delay between nodes in a passively promiscuous mechanism.

Syed et al. showed that clock offset and skew can be corrected in a reliable and efficient manner to achieve time synchronization for mobile underwater networks using the time synchronization for high latency protocol [16]. Two challenges face synchronization of distributed clocks. First, they must be synchronized to a single common event in absolute time or offset (different boot time). Second, one must determine the skew of a given clock relative to some absolute frequency because clocks are imperfect and run at slightly different rates. For the clock offset between two nodes, they can factor out propagation delay via a two-way message (time-stamp) exchange and fix their time difference with one assumption of no clock skewing during the message exchange. For the clock skew, a leading transmitter will send out multiple time-stamped beacons. All receiving nodes will calculate the difference between the received timestamp and the local time, compute a linear regression over all these values, and find the slope of the line. Finally in the second phase offset is found using the skew compensated time.

We have implemented this protocol on the UANT platform (see Fig. 3), which is an extensible software defined underwater acoustic platform [17]. UANT uses GNU Radio and the USRP for the physical layer and uses TinyOS for upper network layers. The application was created to connect PCs forming a network, using Universal software radio peripheral (USRP). The USRP created by Ettus Research [50], is a radio front-end that is commonly used with GNU Radio. Although the option of using a sound card provides a low cost solution, the USRP offers a wider frequency range as well as more dedicated hardware. The USRP has a total of four ADCs and four DACs allowing for up to 16 MHz of bandwidth each way, which is proficient for a custom preamplifier board that also incorporates a switch in order to allow for one transducer per node, as well as amplify the received signal entering the USRP. TinyOS is a widely used sensor network operating system created at University of California, Berkeley and meant for sensor nodes requiring concurrency and flexibility while being



(a) Front view of UANT system (b) Internal view of UANT system

Fig. 3. A deployable UANT system.

limited to resource constraints [51]. TinyOS is implemented in the NesC language, which supports the concurrency model needed for sensor networks. TinyOS is widely used both in commercial applications as well as in academics for research purposes. Fig. 4 shows that after enough beacons are sent the skew between nodes converges, and the nodes share the same notion of time.

Due to clock drift that appears in all oscillators, even after nodes have been synchronized, their clocks will eventually drift apart. This fact leads to the need for periodic resynchronization. The rate at which a synchronization protocol should run largely depends on the properties of the crystals used in oscillators. While inexpensive oscillators tend to have a drift of 30-50 parts per million (ppm), many underwater ranging solutions use more precise clocks (that are temperature compensated) and can achieve accuracies of less than 1 ppm [52]. Two nodes with 50 ppm clocks can accumulate a maximum error of 50 ms in approximately 8.3 minutes, while the clock used by Eustice et al. [52] will accumulate 2 ms of error in just under 14 hours. Therefore, depending on the nodes' hardware, the required resynchronization rate can vary dramatically, but it is still feasible with limited overhead.

Note that to reduce overhead of resynchronization, timestamp information of beacons can be piggybacked in the header of a data packet from the node with the reference clock. In this way when a node is receiving data it can also perform the linear regression and update the values of skew and offset. Since phase two of TSHL requires one packet from the receiving node to be sent back to the transmitter, this information can be appended to the acknowledgement that is sent after the data transfer.

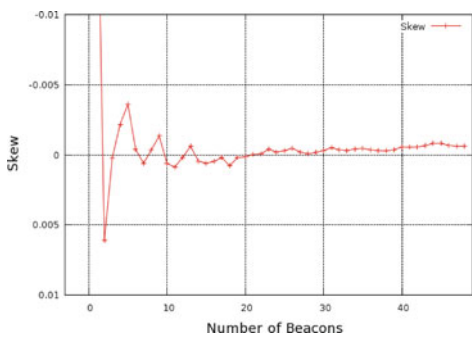


Fig. 4. Number of beacons used in TSHL versus skew estimate.

4 DOTS Design

We now describe our underwater transmission scheduling algorithm, DOTS that exploits long propagation delays by using passively observed one-hop neighboring nodes' transmissions to improve channel utilization. The design of DOTS is based on MACA-like random channel access with RTS/CTS. Because of this design choice, it is confronted with the problem that data transmission between two nearby nodes after RTS/CTS handshaking can be collided with RTS control frames of a distant node due to relatively long propagation delays [53]. Recall that this will happen more frequently and be more expensive in mobile underwater networks than in terrestrial radio networks due to the high latency and transmission costs. Fullmer et al. [54] identified the problem and provided the following two conditions for collision free transmission:

- *RTS wait time* should be greater than the maximum propagation delay that is the propagation delay for a transmitted frame to reach its maximum transmission range.
- *CTS wait time* should be greater than the RTS transmission time plus twice the maximum propagation delay plus the hardware transmit-to-receive transition time.

Thus, these two conditions are the basis of DOTS protocol in order to avoid frame collisions. With the assumption of synchronization, DOTS can locally calculate the distributed transmission and reception schedules to perform concurrent transmissions when viable by promiscuously overhearing neighboring transmissions. DOTS maintains minimal internal states in a delay map database to keep track of observed neighboring transmission and reception schedules. This database is updated based on each observed frame's MAC header. In addition to standard source, destination, sequence number, frame size and cyclic redundancy check (CRC) checksums in the MAC header, DOTS necessitates two additional fields in the MAC header, namely an accurate clock synchronized timestamp of when the frame was sent and an estimate of the propagation delay between the source and destination. This estimate of the propagation delay between the source and the destination of the overheard frame can be performed during the clock synchronization process by examining the time of flight information during the frame exchanges and later updated through further communications between the nodes. Moreover, the

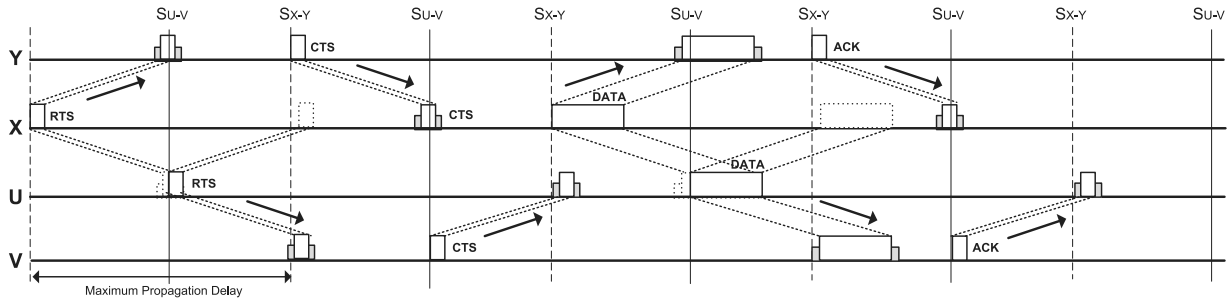


Fig. 5. An example of a concurrent transmission schedule.

delay map database entries can expire and be removed over time with the knowledge of data size of each entry and the maximum propagation delay for each overheard frame in order to keep the number of database entries small.

Whenever a node has a frame to send, it runs a transmission scheduling decision algorithm based on its delay map database to make a decision as to whether or not to begin its transmission, which will be further discussed in Section 4.2. If no conflicts are detected, it begins its transmission; otherwise, it backs off for a random amount of time. It is important to note that unlike traditional CSMA-like protocols, DOTS allows each node to have multiple outstanding packets to receive. Since each node may miss a neighbor's RTS or CTS transmission due to channel fading in underwater, conflict detection schedules may still cause collisions. Thus, to reduce the damage and to avoid deadlock, DOTS provides for a recovery scheme, the details of which will be discussed in Section 4.3. Finally, since deployed nodes are moving along with the ocean current, it requires a guard time to avoid invalid transmission scheduling caused by the node mobility, which will be further discussed in Section 4.4.

4.1 Delay Map Management

By passively observing neighboring transmissions, each node can maintain a *delay map*, which must contain the following information:

- *source*: the sender of the observed MAC frame
- *destination*: the intended destination of the observed MAC frame
- *timestamp*: the time at which the observed MAC frame was sent
- *delay*: the estimated propagation delay between the source and the destination for the MAC frame.

With clock synchronization, the value of the timestamp can not only provide time information for each frame but also be an accurate indicator of the distance between the sender and the overhearing node itself. Each node can calculate a neighbor's propagation delay to itself by subtracting the timestamp of the MAC frame from the reception time of the MAC frame. Thus, the timestamp and delay fields provide additional distance information between the sender and overhearing node and between the sender and intended frame receiver. Given this additional information, each node can build a delay map of its one-hop neighbors and calculate the expected time a response back to the sender of the observed MAC frame will occur.

Due to network dynamics, neighboring nodes' transmissions can be backed-off or canceled. Furthermore, information of delays between each node and its one-hop neighbors can become stale. To adapt to these dynamics, an update process of the delay map is required. Whenever a new transmission is overheard, each node searches the delay map to check for the existence of existing entries based on source and destination fields. When a duplicate entry is detected, the node checks the freshness of the existing item. If the entry is staler than the latter, then the latter replaces the former. As time passes, the delay map may become unnecessarily larger. To keep the size of the delay map manageable, outdated entries are removed. Whenever an entry is added to the delay map, a timer for each entry is set. Once the timeout is triggered for an item, the item is removed from the delay map.

4.2 Transmission Scheduling

Based on the delay map, a node decides whether or not it can transmit a packet without possible interference with a neighbor node's packet reception. To illustrate the advantage of DOTS, consider an exposed terminal topology (y-x u-v) where two nodes, x and u, are exposed to each other. As depicted in Fig. 5, Node x first transmits an RTS destined for node y. Node y replies with a CTS after waiting for the appropriate amount of time (i.e., total packet transmission time + maximum propagation delay). While node y is waiting for CTS transmission, node u also receives this RTS and has data to send. Considering that a collision only occurs in receiver side, it can begin its own transmission to node v concurrently if the following two conditions hold:

- *Neighboring non-interference*. Its current transmission (RTS) and future transmission (DATA) must not interfere with neighbors' ongoing and prospective receptions (node u's prospective RTS and DATA transmissions should not interfere with node x's CTS and ACK receptions).
- *Prospective non-interference*. Its future receptions (CTS and ACK) must not be interfered with by neighbors' prospective transmissions (node u's prospective CTS and ACK receptions should not be interfered with by node x's prospective DATA transmission).

As for the *neighboring non-interference* condition, node u needs to check whether its RTS will interfere with the reception duration of y's CTS at node x or not. The arrival time of y's CTS at node x can be calculated as follows:

$$r_{CTS(y)} = t_{RTS(x)} + \Delta_{CtlProp} + \tau_{CTS(y)} + \Delta_{y \to x}, \quad (1)$$

where $t_{RTS(x)}$ denotes the timestamp of node x's RTS transmission, $\Delta_{CtlProp}$ denotes the sum of the maximum propagation delay between any two nodes and control packet (RTS/CTS/ACK) reception duration, $\tau_{CTS(y)}$ denotes CTS frame processing time, and $\Delta_{y \to x}$ denotes the delay between node y and node x. Reception duration of node y's CTS at node x can be calculated as below:

$$\Delta_{CTS(y)} = \left[r_{CTS(y)}, r_{CTS(y)} + \frac{\ell_{CTS}}{\lambda_{DATA}} \right], \quad (2)$$

where ℓ_{CTS} denotes data length of CTS and λ_{DATA} denotes data rate. Similarly, the arrival time of y's ACK at node x can be calculated as follows:

$$r_{ACK(y)} = r_{CTS(y)} + \Delta_{DataProp} + \tau_{DATA(x)} + \tau_{ACK(y)} + \Delta_{y \to x}, \quad (3)$$

where $\Delta_{DataProp}$ denote the sum of the maximum propagation delay between two nodes and the reception duration for the DATA frame, $\tau_{DATA(x)}$ denotes DATA frame processing time, and $\tau_{ACK(y)}$ denotes ACK frame processing time. Reception duration of y's ACK at node x can be calculated as below:

$$\Delta_{ACK(y)} = \left[r_{ACK(y)}, r_{ACK(y)} + \frac{\ell_{ACK}}{\lambda_{DATA}} \right]. \quad (4)$$

Finally, node u makes a decision to launch its RTS transmission when its *current time + delay from node u to x* is not in the time ranges of (2) and (4).

As for the *prospective non-interference* condition, node u needs to check whether its CTS and ACK reception duration (received from node v) will be interfered with by the reception duration of x's DATA whose intended receiver is y or not. Expected arrival time of v's CTS at node u can be calculated as follows:

$$r_{CTS(v)} = t_{RTS(u)} + \Delta_{CtlProp} + \tau_{CTS(v)} + \Delta_{v \to u}, \quad (5)$$

where $t_{RTS(u)}$ denotes the time-stamp of node u's planned RTS transmission. Reception duration of v's CTS at node u can be calculated as below:

$$\Delta_{CTS(v)} = \left[r_{CTS(v)}, r_{CTS(v)} + \frac{\ell_{CTS}}{\lambda_{DATA}} \right]. \quad (6)$$

We can similarly calculate the expected arrival time of v's ACK at node u as below:

$$r_{ACK(v)} = r_{CTS(v)} + \tau_{DATA(u)} + \Delta_{DataProp} + \tau_{ACK(v)} + \Delta_{v \to u}. \quad (7)$$

Reception duration of v's ACK at node u can be calculated as follows:

$$\Delta_{ACK(u)} = \left[r_{ACK(v)}, r_{ACK(v)} + \frac{\ell_{ACK}}{\lambda_{DATA}} \right]. \quad (8)$$

The expected arrival time of x's DATA at node u can then be calculated as below:

$$r_{DATA(x)} = t_{RTS(x)} + 2 \times \Delta_{CtlProp} + \tau_{CTS(y)} + \tau_{DATA(x)} + \Delta_{x \to u}. \quad (9)$$

Then, the reception duration of x's DATA at node u can be calculated as follows:

$$\Delta_{DATA(x)} = \left[r_{DATA(x)}, r_{DATA(x)} + \frac{\ell_{DATA}}{\lambda_{DATA}} \right]. \quad (10)$$

Now, node u makes a decision to launch its RTS transmission when the time ranges of (6) and (8) is not in (10). Algorithm 1[1] provides a simplified general description of the transmission decision algorithm for the *neighboring non-interference* case. The algorithm for the *prospective non-interference* case can be implemented like the same way of Algorithm 1 based on (6) and (8).

Algorithm 1 Transmission Scheduling Algorithm

```
1: procedure neighboring non-interference(Message m)
2: for all e ∈ delay map entries do
3:   arrival_{m→e.src} ← prop_delay + tx_delay
4:   if e.frame_type == RTS then
5:     arrival_CTS ← e.timestamp + delay_{e.src→e.dest} + prop_ctrl_delay
6:     if arrival_{m→e.src} ∈ [arrival_CTS ± t_GUARD] then
7:       return collision detected
8:     end if
9:     arrival_ACK ← e.timestamp + delay_{e.src→e.dest} + (2 × prop_ctrl_delay) + prop_data_delay
10:    if arrival_{m→e.src} ∈ [arrival_ACK ± t_GUARD] then
11:      return collision detected
12:    end if
13:  else if e.frame_type == CTS then
14:    arrival_DATA ← e.timestamp+delay_{e.src→e.dest}+ prop_ctrl_delay
15:    if arrival_{m→e.src} ∈ [arrival_DATA±t_GUARD] then
16:      return collision detected
17:    end if
18:  else if e.frame_type == DATA then
19:    arrival_ACK ← e.timestamp + delay_{e.src→e.dest} + prop_data_delay
20:    if arriaval_{m→e.src} ∈ [arrival_ACK±t_GUARD] then
21:      return collision detected
22:    end if
23:  else if e.frame_type == ACK then
24:    no check necessary, continue processing
25:  end if
26: end for
27: return no collision detected
28: end procedure
```

4.3 Schedule Recovery

A node may miss its neighbors' RTS/CTS/DATA packets due to the half-duplex nature of the acoustic modem

1. To simplify the pseudo code, $[arrival_{CTS/DATA/ACK}$ + reception duration of CTS/DATA/ACK $\pm t_{GUARD}]$ is abbreviated to $[arrival_{CTS/DATA/ACK} \pm t_{GUARD}]$.

or the lossy nature of an acoustic channel. Under such circumstances, a node may begin its transmission sequence with an incomplete *delay map*, which may cause packet collision. Since each transmission decision is made locally, collision-free scheduling is not guaranteed.

In DOTS, schedule recovery happens at both sender and receiver sides. At the sender side, when an RTS or a DATA frame is sent, a timer is set to the duration by which the corresponding CTS or ACK frame is received. Once this timer expires, the sender realizes that its transmission has been unsuccessful. In either case (i.e., no CTS or ACK reception), the sender will back off and issue a new RTS. DOTS takes a conservative approach of sending a new RTS for the missing ACK to lower the potential damage; i.e., due to an incomplete *delay map* we cannot guarantee safe retransmission of a large packet at that moment. At the receiver side, a packet loss can be detected in a similar fashion when the DATA frame does not arrive before a timer expires. Once the timer expires, the receiver can reset its state either to send frames (if it has any) or to receive future frames.

A node will update its delay map whenever it receives or overhears any packets from its neighbors. Due to packet loss, however, when updating a node's local *delay map*, scheduled Tx/Rx packets at the node may find schedule conflicts with those of neighboring nodes. Conflicts may happen in the following cases; i.e., (1) a node overhears a CTS packet from its neighboring node, but it has failed to overhear the corresponding RTS packet as packet loss has happened or its location is far apart (i.e., located in the same contention domain with the receiver but out of the contention domain of the sender); (2) a node overhears a DATA packet from its neighboring node, but it has failed to overhear the corresponding RTS packet due to packet loss or RTS-CTS packet pair (common contention case). In general, we cannot salvage scheduled Rx packets since a node cannot reschedule Tx packets at the remote nodes under the current protocol model. If there are two or more Rx schedules suffered from a conflict, schedule recovery with multiple senders will be prioritized based on the initial RTS timestamps in a local delay map. Unlike scheduled RX packets, we find that Tx packets at a node can be easily re-scheduled to avoid collision; then, the updated delay map will be used to re-schedule deferred Tx packets.

4.4 Guard Time

DOTS uses a guard time to support node mobility caused by the ocean currents. Each node calculates this guard time as $2 * (average\ movement\ distance/speed\ of\ sound)$ when it checks the transmission scheduling algorithm. The multiplier, 2, is used since both the sender and the receiver may move in opposite directions from each other. This guard time is then added to the guard time in the frame reception duration, which results in a smaller range of allowable concurrent transmissions. Note that by adding guard time, we can also relieve our protocol's sensitivity to non-deterministic packet processing delay in overloaded sensor nodes.

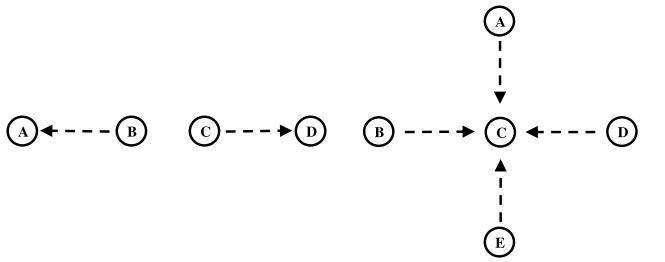


Fig. 6. Simulation topologies.

5 Simulation and Evaluation

5.1 Simulation Setup

5.1.1 Simulation Parameters

For acoustic communications, the channel model described in [55] and [56] 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 the Thorp's formula [56]. As in [55], [57], we use Rayleigh fading to model small scale fading. Unless otherwise mentioned, the data rate is set to 50 kbps as in [58], [59]. We vary data size from 512 bytes to 1 kbyte to observe behavior of each protocol in terms of varying data size. Note again that at a data rate of 50 kbps a 1 kbyte frame requires 0.16384 sec to transmit and the one-way trip delay on a 750 m link is approximately 0.5 sec ($>>$ tx duration $= 0.16384$ sec) considering acoustic propagation delay and transmission duration. We measure throughput and energy consumption per node as the functions of the offered load on the sensor network. The load is varied between generating a single frame every 30 sec down to a single frame every 0.25 sec. In our simulation, each run lasts 1 hour. Unless otherwise specified, we report the average value of 50 runs with the 95 percent confidence interval.

5.1.2 Topology

As shown in Fig 6, we deployed the nodes in a line and a star topologies in a 3D region of $5\ \mathrm{km} \times 5\ \mathrm{km} \times 5\ \mathrm{km}$. In the line topology depicted in Fig. 6a, four nodes are deployed in a line and with a fixed distance between one-hop neighbors. The distance between the nodes are varied from 750 m to 1.5 km for the experiments, and thus the two nodes, B and C, are exposed to each other. We adopt this line topology to show how spatial reuse affects system throughput. As the distance between each pair increases, simulation results will also indicate how temporal reuse can affect system throughput. The star topology, depicted in Fig. 6b, shows a more aggressive traffic toward the center node (c) since the four surrounding nodes attempt to simultaneously send their data to the center node. In this scenario, we create a high contention situation between the four outer nodes for the center node. The distance between the center node and the four surrounding nodes is varied over our experiments ranging from 750 m to 1.5 km. Here, increasing the distance

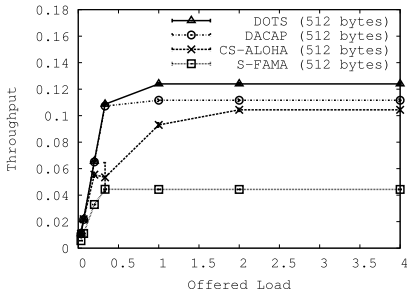


Fig. 7. Line topology: Throughput as a function of offered load with fixed data size (512 bytes).

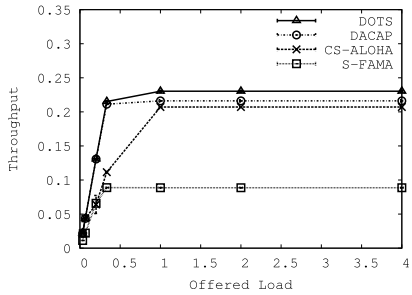


Fig. 8. Line topology: throughput as a function of offered load with fixed data size (1 kbyte).

between the nodes will attest to the benefits of temporal reuse in the presence of high contention.

In addition to these static topologies, we randomly deployed 10 nodes in a 3D region of $430\ \mathrm{m} \times 430\ \mathrm{m} \times 430\ \mathrm{m}$ with a transmission range of 750 m to test node mobility to support the SEA Swarm architecture. This region enables all deployed nodes to be fully connected and exposed to high levels of channel contention as in [18], [49]. We adopt an extended 3D version of the meandering current mobility Model [19] to pattern the motility of each sensor node. Unlike most existing sensor node mobility patterns from literature which assumes 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 sub-surface currents (or jet streams) and vortices on the deployed nodes to pattern its path vector. In our simulations, we restrict the nodes move with a maximum speed of 0.3 m/s with the MCM model to test the resiliency of the guard time in DOTS.

5.2 Simulation Results

5.2.1 Throughput

To evaluate the protocol performance, we measure the throughput as a function of the offered load, defined as follows:

$$\text{Throughput} = \frac{\#\text{ of rx } data \text{ frames} \times \Delta data}{\text{Simulation Duration}}, \tag{11}$$

where Δ_{data} denotes the duration of transmitting a data frame.

$$\text{Offered Load} = \frac{\#\text{ of generated } data \text{ frames} \times \Delta data}{\text{Simulation Duration}}. \tag{12}$$

The performance of DOTS was compared to that of three CSMA protocols, namely Slotted FAMA (S-FAMA) [53], DACAP [60], and CS-ALOHA with ACK [61]. S-FAMA is a synchronized underwater MAC protocol based on RTS/CTS handshaking. The main idea of S-FAMA is to time slot exclusive access to the channel medium so that the time duration of each slot is long enough to ensure that any frame transmitted at the start of the slot will reach the destination before the slot duration ends. DACAP is a non-synchronized protocol that allows each node to use different handshaking lengths for different distances between the sender and the receiver. To reduce collision, DACAP follows these two collision avoidance conditions: 1) when a receiver overhears an RTS threatening its pending data reception, the receiver sends a very short warning frame to its intended sender to defer its data transmission until the predefined waiting period 2) after sending an RTS, if a sender overhears a CTS threatening the neighbor's pending data reception, it defers its data transmission. CS-ALOHA with ACK is ALOHA adapted for the underwater environment, where each node transmits whenever the channel is idle without performing the RTS/CTS handshaking process.

Figs. 7 and 8 show the throughput of the four protocols with different data sizes in the line topology (exposed terminal). As shown in Figs. 7 and 8, DOTS outperforms S-FAMA by a factor of two and DACAP and CS-ALOHA by around 15 percent for a 750 m transmission range with both 512 and $1,024$ byte data frame sizes. It is noteworthy that DACAP outperforms S-FAMA by two times because DACAP allows for concurrent transmissions of the two sender-receiver pairs in Fig. 6a; when a sender-receiver pair (A-B) is undergoing data transmission in the line topology, the other pair (C-D) can also perform parallel data transmission because the two collision avoidance conditions of DACAP cannot suppress the transmissions of the two sender nodes (B and C). Consequently, this allows DACAP to perform concurrent transmissions possibly with collisions; however, it is the result of avoiding these minor collisions which explains the utilization gain of DOTS over that of DACAP. By varying the data size, Figs. 7 and 8 show that data size is proportional to the increase in throughput of all handshaking based protocols. The throughput of CS-ALOHA shows similar throughput performance against DOTS. Although it takes advantage of spatial reuse, it lacks the capability to avoid collisions, thereby offsetting the gains from spatial reuse, which will be addressed in the star topology. It is also interesting to note that the all four protocols show a saturation point. The throughput increases as the offered load increases until a threshold limit. After reaching the threshold point, the all four protocols suppress their transmissions and thus their performance becomes saturated.

In the star topology, the four outer nodes compete to send their frames to the one center node. Figs.9 and 10 show that DOTS outperforms S-FAMA and CS-ALOHA by two times and DACAP by 70 percent for a 750 m transmission range with both 512 and $1,024$ byte data frame sizes. By varying the data size, these two figures show that the three

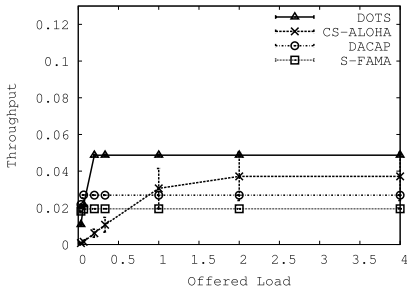


Fig. 9. Star topology: throughput as a function of offered load with fixed data size (512 bytes).

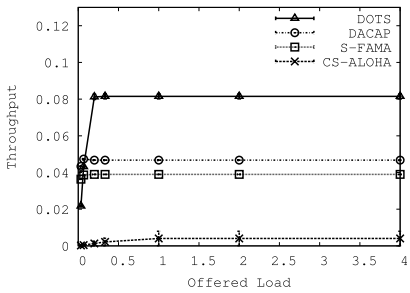


Fig. 10. Star topology: throughput as a function of offered load with fixed data size (1 kbyte).

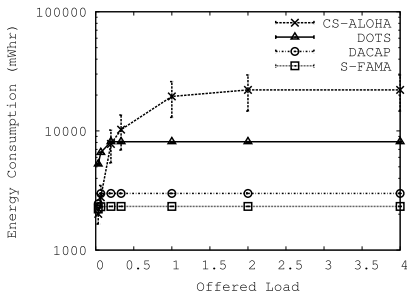


Fig. 11. Star topology: energy consumption in the star topology with fixed data size (512 bytes).

handshaking based protocols exhibit the behavior that throughput is proportional to data frame size. On the other hand, CS-ALOHA shows unstable throughput performance; when the data size exceeds a threshold, CS-ALOHA significantly increases its collision rate and reduces its overall throughput. In contrast, DOTS shows a vastly superior behavior. As the number of senders increases, DOTS can better exploit temporal reuse. In this star topology, DOTS outperforms S-FAMA by two times and DACAP by 70 percent. Inversely, CS-ALOHA provides the worst throughput due to absence of collision avoidance.

5.2.2 Energy Consumption

Fig. 11, which represents the four throughput lines of the protocols in Fig. 9, shows the average power consumption of the four protocols in the star topology with a 750 m transmission range and 1,024 byte data frame size. It shows the average energy consumption of each protocol per node during the entire simulation. When it is compared with the throughput lines of the four protocols in Fig. 9, it implicitly indicates that the number of collisions which occur in each protocol. DOTS consumes more energy than S-FAMA and DACAP because it delivers, by far, more frames than these two protocols. Inversely, throughput for CS-ALOHA about 20 percent lower than that of DOTS, yet the energy consumption of CS-ALOHA is several times higher illustrating that CS-ALOHA consumes significantly more energy due to collisions.

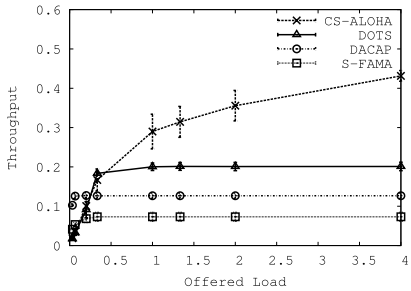


Fig. 12. MCM scenario: Throughput as a function of offered load with fixed data size (512 bytes).

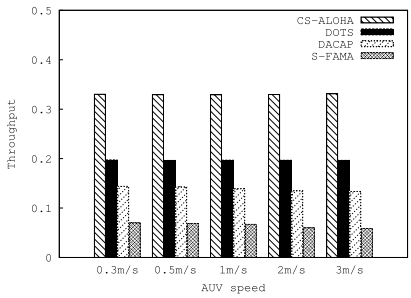


Fig. 13. Throughput according to different MCM mobility speeds.

5.2.3 Impact of MCM Mobility

The effect of MCM mobility is examined in Figs. 12 and 13. Ten nodes are randomly deployed to a region which enables full connectivity between all nodes, whereby each node follows a jet stream path vector based on the MCM model. The main jet stream speed of each node is capped at 0.3 m/s with each node having a 750 m transmission range. Five pairs of sender-receiver nodes are actively engaged in data communication, transmitting 512 byte data packets. Note that with a 0.3 m/s jet stream, nodes can move approximately 20 m in 60 sec, henceforth a 20 ms guard time is amply chosen for use in DOTS to allow for approximately up to a 30 m variation of node locality.

Fig. 12 shows that DOTS outperforms DACAP by 30 percent and S-FAMA by three times. With MCM mobility model, DOTS clearly provides reliable throughput and performance gains over DACAP and S-FAMA by utilizing smart and adaptive scheduling techniques to harness temporal and spatial reuse. CS-ALOHA shows the best performance in the random topologies with node mobility for our test parameters, but, this comes at a steep price of energy efficiency and fairness, which will be addressed later in this section. To understand the impact of speed, we vary the maximum speed of mobile nodes from 0.3 to 3 m/s and plot the maximum throughput under different mobility in Fig. 13. For DOTS, each node's guard time is set based on the guard time equation; the higher the speed, the longer the guard time interval. Recall that the guard time is used to make protocols more robust to mobility (for transmission scheduling with an imprecise

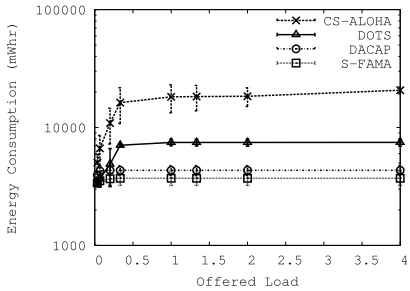


Fig. 14. Energy consumption in the MCM scenario with fixed data size (512 bytes).

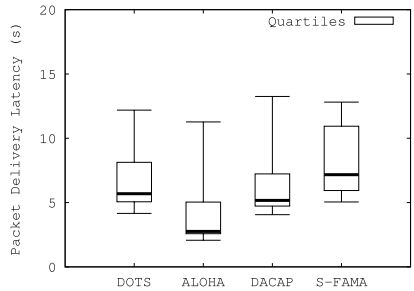


Fig. 16. Line topology: Packet deliver latency with fixed data size (512 bytes).

delay map). The figure confirms that mobility does not cause any significant throughput changes to DOTS under the fully connected scenarios.

Fig. 14 shows the average energy consumption of the four protocols with 750 m transmission range and 512 byte data frame size. The overall trend of this MCM scenario is quite consistent with the previous results in Fig. 11. Note that DOTS consumes more energy than S-FAMA and DACAP because it delivers by far more frames than these two protocols. In the same way, the energy consumption of CS-ALOHA is several times higher than other protocol because of increased number of received packets and collisions.

5.2.4 Guard Time

Evaluating the performance of DOTS by varying the guard time intervals is important as we can show the sensitivity of guard time with respect to the speed of nodes. If the guard time is too short, the chances of packet collisions will be too high. If it is too long, packet collisions will rarely happen, but we have lower chances of exploiting temporal/spatial reuse. In Fig. 15, we show the throughput performance based on different guard time intervals ranging from 1 to 8 ms. All intervals show positive correlation with offered load. It shows that the guard time interval of 2 ms shows the best throughput performance. The guard time intervals of 1 and 8 ms show slightly lower throughput performance due to collisions and lower utilization, respectively.

5.2.5 Packet Delivery Latency

We compare the overall latency for packet delivery that includes RTS/CTS exchanges, data delivery, and ACK reception. We measure the latency by analyzing the packet transmission log data. For latency measurement, we search for the first RTS packet of a successfully delivered data packet; thus, considering channel contention and packet loss. For fair comparison of different protocols, we did not include queueing delay; DOTS is more favorable in terms of queueing delay than the other protocols. To understand the impact of topologies, we use two scenarios, i.e., line and star topologies.

In the line topology, we deployed two nodes in a line with a fixed distance of 1.5 km. In this topology, one sender transmits 512 byte data frames to the intended receiver. Fig. 16 shows the overall latencies of the four protocols. The box plot describes the central tendency of the latency in terms of the median of the values, represented by the smallest box in the plot. The spreads (i.e., variability) of the latency values are represented by quartiles (the 25th and 75th percentiles, a larger box in the plot). The minimum and maximum values of the latency are represented by *whiskers* in the plot. The minimum latency of DOTS is 4.16 s and is composed of Tx and Rx durations of RTS, CTS, DATA, and ACK, 4×propagation delay between two sender and receiver for each transmission, and 4×guard time for each reception. Since Aloha does not have RTS and CTS handshakes, its minimum latency is given as 2.07 s which is composed of Tx and Rx durations of DATA and ACK and 2×propagation delay between the sender and receiver. The minimum latency of S-FAMA is given as 5.05 s which is composed of Tx and Rx durations of RTS, CTS, DATA, ACK and 4×propagation delay between the sender and receiver. According to [48], each packet (RTS, CTS, DATA or ACK) has to be transmitted at the beginning of a slot and the slot length has to be $\tau + \gamma$, where τ is the maximum propagation delay and γ is the Tx duration of a CTS packet. Thus, S-FAMA requires one more slot to complete the DATA Tx because DATA Tx duration exceeds the CTS Tx duration, thereby requiring an additional slot to complete the DATA Tx. This is why S-FAMA shows the worst minimum latency in the figure. The minimum latency of DACAP is reported to be 4.06 s which is composed of Tx and Rx durations of RTS, CTS, DATA, ACK and 4×propagation delay between the sender and receiver.

As depicted in Fig. 6b, we also measure the latencies of the four protocols in a star topology where the four surrounding nodes attempt to simultaneously send their packets to the center node, and the distance between the sender and center node is 1.5 km. In this high contention scenario, each sender transmits 512 byte data frames to the receiver. Fig. 17 shows the latencies of the four protocols.

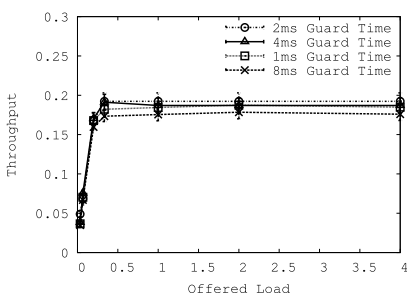


Fig. 15. Guard time sensitivity to a MCM mobility speed (3 m/s).

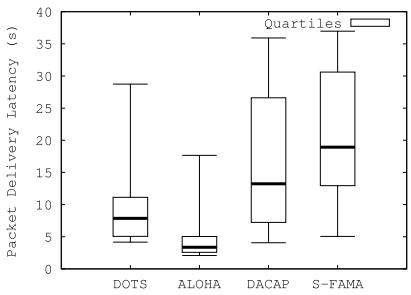


Fig. 17. Star topology: Packet deliver latency with fixed data size (512 bytes).

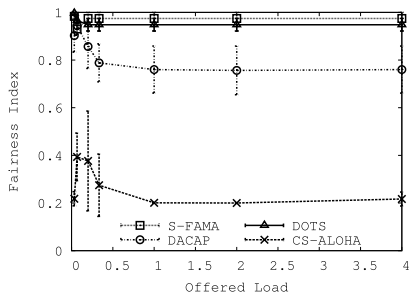


Fig. 18. Jain's fairness index for the four protocols.

The averaged latency of DOTS is superior to DACAP and S-FAMA because DOTS exploits spatial and temporal reuse and allows a more number of on-the-fly packets. As shown in Fig. 9, DOTS's latency is given as 8.86 s on average, whereas other protocols have much higher latencies. DACAP's latency is 16.23 s on average and it outperforms S-FAMA whose average latency is 20.93 s. This gain is due to DACAP's capability of spatial reuse. As in the line topology, CS-ALOHA's performance is far superior to other protocols as it lacks channel reservation, resulting the latency of 3.86 s. However, as shown earlier, this gain comes at the cost of low data rate and protocol fairness (due to high collision and lack of fairness control).

5.2.6 Fairness

MAC protocols with backoff schemes (i.e., binary exponential) based on insufficient information about the network congestion may cause *spatial unfairness*, a form of channel capture, as described in Syed et al. [18]. Since a frame's propagation latency is proportional to the distance from a sender, the channel clears earlier for nodes closer to the sender. Closer nodes consequently have more opportunities to recapture the channel, resulting in unfairness amongst the nodes. To characterize the fairness, we use the Jain Fairness Index [62], defined as below

$$\text{Fairness Index} = \frac{(\sum x_i)^2}{(n \cdot \sum x_i^2)}, \tag{13}$$

where x_i denotes the throughput of node i and n denotes the number of nodes in the network. Fig. 18, which is the corresponding fairness plot to Fig. 13, shows that S-FAMA and DOTS exhibit a high fairness index (0.9 and above) and also remain stable and constant with increased offered load. As described in 4.3, when more than one transmission schedule contends in a node, DOTS uses the timestamp knowledge in its delay map database to give preference to one of the transmission schedules. DOTS with random backoff exhibits high fairness for this reason. The reason for the slightly lower fairness of DOTS compared to S-FAMA is due to the use of temporal and spatial reuse. In DOTS, every sender-receiver pair has a fair chance of accessing the medium as in S-FAMA, yet some pairs are given the chance of concurrently accessing the medium, thus slightly affecting the fairness index. DACAP provides a lower fairness index than both S-FAMA and DOTS. This is because DACAP gives priority to the nodes already accessing the channel and consequently causes this bias. CS-ALOHA shows the lowest fairness index and the largest variation. Due to CS-ALOHA's binary exponential backoff, it allows close sender-receiver pairs to potentially capture the channel, thereby severely degrading the fairness but providing best throughput performance as indicated in Fig. 12. This channel capturing also leads to severe data collisions at other nodes which have not captured the channel, inducing poor energy utilization. Furthermore, as Fig. 12 indicates CS-ALOHA is subject to far greater amounts of instability and throughput variation as a result of this capture effect.

6 Towards Enabling Multiple Transmission Sessions

We discuss our preliminary study on enabling multiple transmission sessions in DOTS. Here, the term session refers to opening, closing, and managing a communications dialogue between end-user application processes (i.e., a sequence of RTS-CTS-DATA-ACK packet exchanges between a sender and its intended receiver). In the original DOTS, a receiver opportunistically has permitted multi-session reception whenever an incoming RTS does not conflict with its current ongoing transmission session. However, a sender can have only one outstanding packet (i.e., single transmission session but multiple reception sessions). Given that there could be opportunities of having multiple outstanding packet transmissions, we investigate a mechanism of scheduling multi-session transmission (called MDOTS). We allow each node to manage multiple independent sessions, and thus, there could be multiple outstanding packets within a session period (pipelined).

To illustrate the advantages of MDOTS over DOTS, consider a line topology ($A - B - C$) where node B can reach both A and C, but they are hidden from one another. In this case, DOTS is only able to transfer one session (i.e., RTS-CTS-DATA-ACK sequence) to a single receiver. As depicted in Fig. 19, in the same amount of time, MDOTS can actively initiate two different sessions (one to node A and the other to C); node B first transmits an RTS destined for node A. While the RTS packet is still propagating, node B waits for a random period of time and then transmits another RTS destined for node C. When node A receives its RTS, it waits until time has passed (i.e., total packet transmission time + maximum propagation delay) and then replies with a CTS. Meanwhile, node C has also received its RTS, and replies with a CTS after waiting the appropriate amount of time. Node B receives CTS messages from both nodes A and C

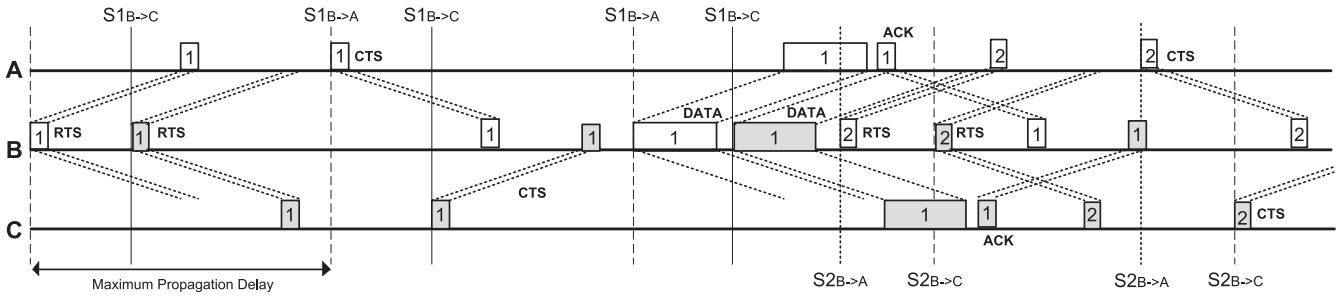


Fig. 19. Enabling multiple transmission sessions.

sequentially and then sends the DATA packets respectively. Note that node B can initiate multiple sessions to each destination node (either A or C) by sending another RTS to the destination before receiving an ACK from nodes A or C for the previous session's DATA transmissions.

To support this, the MAC layer is extended to maintain a buffer that queues multiple packets received from the Network layer. It allows us to maintain multiple concurrent packet transmissions. MDOTS can easily support out of packet delivery in case of head of line blocking. Consider a case where node B has two packets in a buffer (the first destined to node A and the second destined to node C). Assuming that node B cannot transmit a packet to node A due to expected collision, it can immediately start a new session with node C. The scheduling process is the same as DOTS. When scheduling multiple packet transmissions, a sender must track the number of outstanding transmission sessions and simultaneously check expected collisions with the delay map. To avoid a single node capturing (or monopolizing) the whole channel, we simply limit the maximum number of transmission sessions on the fly per node. In our study, we vary the maximum number from 2, 4, 8, and 16 (denoted as MDOTS-2, -4, -8, and -16). We are currently investigating a mechanism of dynamically adjusting the number of sessions for fair bandwidth sharing (and permitting variable packet size and packet training).

To understand the performance benefits of MDOTS, we perform preliminary simulations in two representative topologies namely the topology of 4 Senders and 1 Sink (Fig. 6b), and the topology of 1 Sender and 4 Sinks. We expect that the topology of 1 Sender and 4 Sinks is more favorable to M-DOTS variants because the sender does not compete with other neighbors to access channel but can maintain the maximum number of outstanding sessions to the sinks. Unless otherwise mentioned, we set the distance between Sender and Sink as 750 m and the size of packets as 512bytes.

Four Senders and One Sink: Fig. 20 shows the throughput of the five protocols in 4-Senders 1-Sink topology (shown in Fig 6b) when data packet size is 512 byte and transmission range is 750 m. For this case, MDOTS-2 shows the best throughput performance. However, MDOTS protocols with more than two sessions show degraded throughput than original DOTS protocol. With this topology, four senders are competing with each other. None of the senders are close enough to directly overhear each other's RTS message. A sender will never hear a RTS from another sender, which means that senders cannot find out ongoing sessions between the sink and another sender until it overhears the CTS from the sink. Since the senders do not estimate the degree of congestion, aggressive protocols like MDOTS-4, MDOTS-8, and MDOTS-16 will tend to be overly aggressive in sending RTS messages trying to capture the sink's attention. The sink will be overwhelmed with many RTS messages from the four senders. This partly explains the lower performance results of MDOTS-8 and MDOTS-16. To alleviate and provide reliable throughput performance, we need a mechanism to accurately measure contending senders in a contention domain and allow promising new sessions, which remains as our future work.

1 Sender and 4 Sinks: Fig. 21 shows the throughput of the five protocol in the topology of 1 Sender and 4 Sinks with data packet size of 512 byte and transmission range of 750 m. In this scenario, all MDOTS protocols outperform original DOTS protocol. Amongst the MDOTS variants, MDOTS-16 shows the best performance. The 4-topology is similar in layout to the topology of 4 Senders and 1 Sink, but the roles of Sender and Sink are switched.

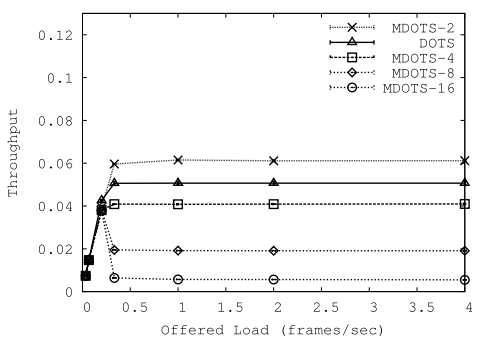


Fig. 20. 4-Senders 1-Sink: Throughput with fixed data size (512 bytes).

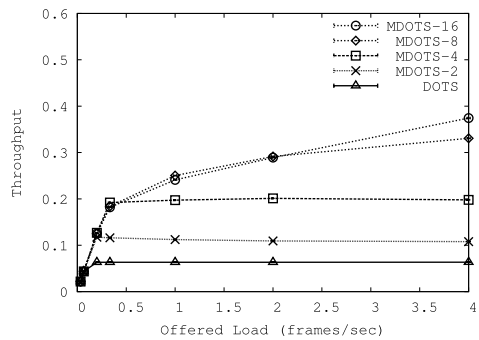


Fig. 21. 1-Sender 4-Sinks: Throughput with fixed data size (512 bytes).

Instead of four outer nodes all trying to start sessions with a central node, one central node tries to initiate sessions with the 4 Sinks. This configuration works very well with aggressive versions of MDOTS (e.g., MDOTS-16). In this topology, a central node is a participant of every communication, so it always has full and current knowledge of all ongoing communication sessions. The delay map and collision avoidance of MDOTS protocols allow the central sender to leverage its full knowledge and schedule transmission sessions without any collisions, leading to better throughput. The results show that MDOTS-16 achieves the saturated throughput performance.

7 Conclusions

We proposed a MAC protocol called DOTS that alleviates limitations caused by the long propagation latency and the severely limited bandwidth of acoustic communications. DOTS aimed to achieve better channel utilization by harnessing both temporal and spatial reuse. Extensive simulation results showed that (1) DOTS outperforms S-FAMA by 200 percent and DACAP by 15 percent in the line topology (exposed terminal) and S-FAMA by 200 percent and DACAP by 70 percent in the star topology (higher node density and contention), and (2) DOTS provides reliable throughput performance even with node mobility and preserves a high level of fairness for channel access. Moreover, we have introduced a mechanism of enabling multiple transmission sessions (called MDOTS). A preliminary evaluation showed that MDOTS significantly outperforms the original DOTS.

Acknowledgments

This work was funded by the MSIP (Ministry of Science, ICT & Future Planning), Korea in the ICT R&D Program 2013 and also supported in part by the US National Science Foundation under Grant No. 1205757.

References

[1] S. Roy, P. Arabshahi, D. Rouseff, and W. Fox, "Wide Area Ocean Networks: Architecture and System Design Considerations," *Proc. First ACM Int'l Workshop on Underwater Networks (WUWNet'06)*, Sept. 2006.

[2] I.F. Akyildiz, D. Pompili, and T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges," *Ad Hoc Networks*, vol. 3, pp. 257-279, 2005.

[3] 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.

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

[5] J. Jaffe and C. Schurgers, "Sensor Networks of Freely Drifting Autonomous Underwater Explorers," *Proc. First ACM Int'l Workshop on Underwater Networks (WUWNet '06)*, 2006.

[6] U. Lee, J. Kong, J.S. Park, E. Magistretti, and M. Gerla, "Time-Critical Underwater Sensor Diffusion with No Proactive Exchanges and Negligible Reactive Floods," *Proc. IEEE 11th Symp. Computers and Comm. (ISCC '06)*, 2006.

[7] Z. Zhou, J. Cui, and A. Bagtzoglou, "Scalable Localization with Mobility Prediction for Underwater Sensor Networks," *IEEE Trans. Mobile Computing*, vol. 10, no. 3, pp. 335-348, Mar. 2011.

[8] *Micro-Modem Overview*, Woods Hole Oceanographic Institution, http://acommn.whoi.edu/micromodem, 2014.

[9] E. Gallimore, J. Partan, I. Vaughn, S. Singh, J. Shusta, and L. Freitag, "The WHOI Micromodem-2: A Scalable System for Acoustic Communications and Networking," *Proc. Oceans'10*, Sept. 2010.

[10] R.J. Urick, *Principles of Underwater Sound*. third ed., McGraw-Hill, 1983.

[11] X. Guo, M. Frater, and M. Ryan, "A Propagation-Delay-Tolerant Collision Avoidance Protocol for Underwater Acoustic Sensor Networks," *Proc. Asia Pacific OCEANS*, 2006.

[12] J. Yackoski and C.-C. Shen, "UW-FLASHR: Achieving High Channel Utilization in a Time-Based Acoustic MAC Protocol," *Proc. Third ACM Int'l Workshop Underwater Networks (WUWNet)*, 2008.

[13] C. Hsu, K. Lai, C. Chou, and K.C. Lin, "ST-MAC: Spatial-Temporal MAC Scheduling for Underwater Sensor Networks," *Proc. IEEE INFOCOM*, 2008.

[14] K. Kredo, P. Djukic, and P. Mohapatra, "STUMP: Exploiting Position Diversity in the Staggered TDMA Underwater MAC Protocol," *Proc. IEEE INFOCOM*, 2009.

[15] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A Media Access Protocol for Wireless LAN's," *Proc. SIGCOMM*, 1994.

[16] A.A. Syed and J. Heidemann, "Time Synchronization for High Latency Acoustic Networks," *Proc. IEEE INFOCOM*, 2006.

[17] D. Torres, J. Friedman, T. Schmid, and M.B. Srivastava, "Software-Defined Underwater Acoustic Networking Platform," *Proc. Fourth ACM Int'l Workshop UnderWater Networks (WUWNet)*, 2009.

[18] A.A. Syed, W. Ye, and J. Heidemann, "T-Lohi: A New Class of MAC Protocols for Underwater Acoustic Sensor Networks," *Proc. IEEE INFOCOM*, 2008.

[19] 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.

[20] Y. Noh, P. Wang, U. Lee, D. Torres, and M. Gerla, "DOTS: A Propagation Delay-Aware Opportunistic MAC Protocol for Underwater Sensor Networks," *Proc. IEEE Int'l Conf. Network Protocols (ICNP)*, 2010.

[21] B.M. Howe, T. McGinnis, and J. Gobat, "Moorings for Ocean Observatories: Continuous and Adaptive Sampling," *Proc. Scientific Submarine Cable Conf.*, Feb. 2006.

[22] J. Rice, "SeaWeb Acoustic Communication and Navigation Networks," *Proc. Underwater Acoustic Measurements*, July 2005.

[23] W.J. Gould and W.J. Gould, "Argo—Sounding the Oceans," *Weather*, vol. 61, no. 1, pp. 17-21, Jan. 2006.

[24] J. Bellingham, *Autonomous Underwater Vehicles (AUVs)*. Academic Press, 2001.

[25] D.B. Kilfoyle and A.B. Baggeroer, "The State of the Art in Underwater Acoustic Telemetry," *IEEE J. Oceanic Eng.*, vol. 25, no. 1, pp. 4-27, Jan. 2000.

[26] D. Bertsekas and R. Gallager, *Data Networks*. Prentice Hall, 1992.

[27] D. Pompili, T. Melodia, and I.F. Akyildiz, "A CDMA-Based Medium Access Control Protocol for Underwater Acoustic Sensor Networks," *IEEE Trans. Wireless Comm.*, vol. 8, no. 4, pp. 1899-1909, Apr. 2009.

[28] P. Xie and J.H. Cui, "R-MAC an Energy-Efficient MAC Protocol for Underwater Sensor Networks," *Proc. Int'l Conf. Wireless Algorithms, Systems and Applications (WASA)*, 2007.

[29] M. Chen, S. Gonzalez, and V. Leung, "Applications and Design Issues for Mobile Agents in Wireless Sensor Networks," *IEEE Wireless Comm.*, vol. 14, no. 6, pp. 20-26, Dec. 2007.

[30] H.-H. Ng, W.-S. Soh, and M. Motani, "Maca-u: A Media Access Protocol for Underwater Acoustic Networks," *Proc. IEEE Global Telecomm. Conf. (GLOBECOM)*, 2008.

[31] P. Karn, "MACA : A New Channel Access Protocol for Packet Radio," *Proc. Ninth Computer Networking Conf. ARRL/CRRL Amateur Radio*, pp. 134-140, Sept. 1990.

[32] N. Chirdchoo, W.Seng Soh, and K. Chua, "RIPT: A Receiver-Initiated Reservation-Based Protocol for Underwater Acoustic Networks," *IEEE J. Selected Areas in Comm.*, vol. 26, no. 9, pp. 1744-1753, Dec. 2008.

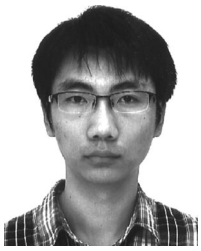
[33] N. Chirdchoo, W.-S. Soh, and K.C. Chua, "Maca-MN: A Maca-Based MAC Protocol for Underwater Acoustic Networks with Packet Train for Multiple Neighbors," *Proc. IEEE Vehicular Technology Conf.*, 2008.

[34] K. Kredo II, and P. Mohapatra, "Scheduling Granularity in Underwater Acoustic Networks," *Proc. Sixth ACM Int'l Workshop Underwater Networks (WUWNet '11)*, 2011.

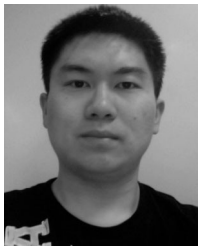
[35] H.-H. Ng, W.-S. Soh, and M. Motani, "ROPA: A MAC protocol For Underwater Acoustic Networks with Reverse Opportunistic Packet Appending," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC)*, 2010.
[36] H.-H. Ng, W.-S. Soh, and M. Motani, "BiC-MAC: Bidirectional-Concurrent MAC Protocol with Packet Bursting for Underwater Acoustic Networks," *Proc. OCEANS Conf.*, 2010.
[37] A. Acharya, A. Misra, and S. Bansal, "Design and Analysis of a Cooperative Medium Access Scheme for Wireless Mesh Networks," *Proc. First Int'l Conf. Broadband Networks (BroadNets '04)*, 2004.
[38] K. Mittal and E. Belding, "RTSS/CTSS: Mitigation of Exposed Terminals in Static 802.11-Based Mesh Networks," *Proc. IEEE Second Workshop on Wireless Mesh Networks (WiMesh)*, 2006.
[39] M. Cesana, D. Maniezzo, P. Bergamo, and M. Gerla, "Interference Aware (IA) MAC: An Enhancement to IEEE802.11b DCF," *Proc. IEEE 58th Vehicular Technology Conf. (VTC)*, 2003.
[40] D. Shukla, L. Chandran-Wadia, and S. Iyer, "Mitigating the Exposed Node Problem in IEEE 802.11 Ad Hoc Networks," *Proc. 12th International Conf. Computer Comm. and Networks (ICCCN)*, 2003.
[41] S. Eisenman and A. Campbell, "E-CSMA: Supporting Enhanced CSMA Performance in Experimental Sensor Networks Using Per-Neighbor Transmission Probability Thresholds," *Proc. IEEE INFOCOM*, 2007.
[42] R. Diamant, G. Shirazi, and L. Lampe, "Robust Spatial Reuse Scheduling in Underwater Acoustic Communication Networks," *Proc. IEEE Vehicular Technology Conf.*, 2011.
[43] J. Ma and W. Lou, "Interference-Aware Spatio-Temporal Link Scheduling for Long Delay Underwater Sensor Networks," *Proc. IEEE Eighth Ann. Comm. Soc. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON)*, 2011.
[44] R. Diamant, W. Shi, W. Soh, and L. Lampe, "Joint Time and Spatial Reuse Handshake Protocol for Underwater Acoustic Communication Networks," *Proc. OCEANS Conf.*, 2011.
[45] A. Acharya, A. Misra, and S. Bansal, "Maca-p: A MAC for Concurrent Transmissions in Multi-Hop Wireless Networks," *Proc. First IEEE Int'l Conf. Pervasive Computing and Comm. (PerCom '03)*, 2003.
[46] M. Vutukuru, K. Jamieson, and H. Balakrishnan, "Harnessing Exposed Terminals in Wireless Networks," *Proc. Fifth USENIX Symp. Networked Systems Design and Implementation (NSDI)*, 2008.
[47] L.G. Roberts, "ALOHA Packet System with and without Slots and Capture," *ACM SIGCOMM Computer Comm. Rev.*, vol. 5, pp. 28-42, Apr. 1975.
[48] M. Molins and M. Stojanovic, "Slotted FAMA: A MAC Protocol for Underwater Acoustic Networks," *Proc. OCEANS Conf.*, 2006.
[49] A. Syed, W. Ye, and J. Heidemann, "Comparison and Evaluation of the T-Lohi MAC for Underwater Acoustic Sensor Networks," *IEEE J. Selected Areas in Comm.*, vol. 26, no. 9, pp. 1731-1743, Dec. 2008.
[50] "USRP Brochure," http://www.ettus.com/, Apr. 2009.
[51] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, E.B.M. Welsh, and D. Culler, *Ambient Intelligence*, pp. 115-148, Springer, 2005.
[52] R. Eustice, L. Whitcomb, H. Singh, and M. Grund, "Recent Advances in Synchronous-Clock One-Way-Travel-Time Acoustic Navigation," *Proc. OCEANS Conf.*, 2006.
[53] C.L. Fullmer and J.J. Garcia-Luna-Aceves, "Floor Acquisition Multiple Access (FAMA) for Packet-Radio Networks," *Proc. ACM SIGCOMM*, 1995.
[54] C.L. Fullmer and J.J. Garcia-Luna-Aceves, "Solutions to Hidden Terminal Problems in Wireless Networks," *Proc. ACM SIGCOMM*, 1997.
[55] M. Stojanovic, "On the Relationship Between Capacity and Distance in an Underwater Acoustic Communication Channel," *SIGMOBILE Mobile Computing Comm. Rev.*, vol. 11, pp. 11-22, 2007.
[56] P.L.L.M. Brekhovskikh and Yu, *Fundamentals of Ocean Acoustics*. third ed., Springer, 2003.
[57] C. Carbonelli and U. Mitra, "Cooperative Multihop Communication for Underwater Acoustic Networks," *Proc. First ACM Int'l Workshop on Underwater Networks (WUWNet)*, 2006.
[58] Z. Zhou and J.-H. Cui, "Energy Efficient Multi-Path Communication for Time-Critical Applications in Underwater Sensor Networks," *Proc. MobiHoc*, 2008.
[59] 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.
[60] B. Peleato and M. Stojanovic, "Distance Aware Collision Avoidance Protocol for Ad-Hoc Underwater Acoustic Sensor Networks," *IEEE Comm. Letters*, vol. 11, no. 12, pp. 1025-1027, Dec. 2007.
[61] C.L. Fullmer and J.J. Garcia-Luna-Aceves, "Analysis of Aloha Protocols for Underwater Acoustic Sensor Networks," *Proc. Workshop UnderWater Networks (WUWNet)*, 2006.
[62] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems,"technical report DEC Research, 1984.

Young Tae Noh received the BS degree in computer science from Chosun University in 2005, the MS degree in information and communication from Gwangju Institute of Science Technology in 2007, and the PhD degree in computer science at University of California, Los Angeles, in 2012. He is currently a software engineer at Cisco Systems, Inc. His research areas include data center networking, wireless networking, future Internet, and mobile/pervasive 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 at Los Angeles in 2008. He is an assistant professor in the Department of Knowledge Service Engineering at Korea Advanced Institute of Science and Technology. Before joining KAIST, he was a member of 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.



Seongwon Han is currently working toward the PhD degree in the Network Research Lab at the University of California, Los Angeles (UCLA) under the guidance of Dr. Mario Gerla. He received the BS degree in computer science from Ajou University in 2006 and the MS degree in computer science from UCLA in 2009. His primary research interests are Media Access Control (MAC), routing protocols and localization in underwater sensor networks. He is a student member of the IEEE.



Paul Wang received the BS degree 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 currently a senior software engineer at Morgan Stanley & Co. LLC. Prior to joining Morgan Stanley, he was a software engineer at Knight Capital Group, Inc. and a systems software engineer at NASA's Jet Propulsion Laboratory (JPL). 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.

Dustin Torres received the BS degree in computer engineering from the University of California, Irvine, in 2008 and the MS degree in electrical engineering from the University of California, Los Angeles, in 2010. He is a firmware engineer for Intel Corporation. His research interests include wireless sensor networks and underwater wireless communication. He is a member of the IEEE.

Jinwhan Kim received the BS and MS degrees in naval architecture and ocean engineering from Seoul National University, Seoul, Korea, in 1993 and 1995. Then, he received the MS and the PhD degrees in aeronautics and astronautics from Stanford University in 2002 and 2007, respectively. From 1995 to 1999, he was with Korea Institute of Machinery and Materials as a research engineer, and subsequently with Korea Ocean Research and Development Institute as a senior research engineer until 2000. He was also with Optimal Synthesis Inc. as a research scientist from 2007 until 2010. He is currently an assistant professor in the Division of Ocean Systems Engineering at KAIST, Daejeon, Korea, working in the areas of robotics, vehicle dynamics, control and estimation. He is a member of the IEEE, and a senior member of AIAA.

Mario Gerla received the engineering degree from Politecnico di Milano, Italy, and the PhD degree from UCLA. He is a professor in the Department of Computer Science at UCLA. At UCLA, he was 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 (see www.cs.ucla.edu/NRL for recent publications). He has been a fellow of the IEEE since 2002.