# The Underwater Selective Repeat Error Control Protocol for Multiuser Acoustic Networks: Design and Parameter Optimization

Saiful Azad, Paolo Casari, *Member, IEEE*, Michele Zorzi, *Fellow, IEEE*

***Abstract*—In this paper, we introduce Underwater Selective Repeat (USR), a Selective Repeat Automatic Repeat reQuest (SR-ARQ) mechanism for multiuser underwater acoustic networks. Our scheme exploits the typically large round-trip time (RTT) of underwater acoustic links to interlace the transmission of data and acknowledgment (ACK) packets, such that the transmitter never starts sending data packets when it should receive ACKs. No specific synchronization mechanism is required to do so. It is shown that the timing of point-to-point communications can be adjusted to optimize the performance of multiuser networks of a given size. Moreover, it is shown that the proposed strategy can be made robust to mobility, hence to time-varying RTTs.**

**We provide detailed simulation results that assess the performance of USR as a function of the protocol parameters, both in static and in mobile networks. Based on these results, we propose an adaptive version of USR, whereby a node can modify its behavior (e.g., it can pack data transmissions more tightly or more loosely within one RTT) by reacting to packet errors induced by multiple-access interference.**

***Index Terms*—Underwater networks, selective repeat ARQ, multiuser transmissions, simulation, WOSS, parameter optimization.**

## I. Introduction and Related Work

ERROR control techniques are of prominent importance to counter reception errors in underwater (UW) acoustic communications [1]. Usually, a Forward Error Correction (FEC) code is natively implemented in the physical layer (PHY) of commercial UW communication devices. Nevertheless, movement- or environment-induced fluctuations of the signal power at the receiver, as well as the effect of multiple access interference in multiuser networks, can generate detection errors that exceed the correction capabilities of the PHY-level FEC. In this case, Automatic Repeat reQuest (ARQ) policies are needed to recover from packet errors. Such events are quite likely in real scenarios: for this reason, pioneering efforts in UW networking such as SeaWeb [2] incorporate some form of ARQ.

Generally speaking, ARQ schemes prescribe that the receiver send one acknowledgment packet (ACK) back to the transmitter for every data packet correctly received. In case of errors (e.g., as detected via a failed CRC check) the receiver transmits one not-Acknowledged (NACK) packet, or remains silent. This allows the sender to detect the reception error and to take action by retransmitting erroneous packets. The policy most typically employed to administer retransmissions is a form of Stop-and-Wait (S&W) ARQ [3], both for its simplicity and because S&W can be straightforwardly employed over half-duplex media such as the UW acoustic channel. S&W ARQ prescribes that the sender transmit a data packet and wait for the corresponding ACK/NACK packet, before sending the next data packet. If no ACK is received within a timeout period, or a NACK packet is received, the corresponding data packet is transmitted again. This choice is inherently inefficient for UWANs, as it requires a sender to remain idle for one whole round-trip time (RTT). As the propagation speed of the acoustic waves underwater is low, the throughput achieved by S&W is limited [4]. Therefore, several protocols were designed to perform multiple packet transmissions back-to-back [4], [5], hence achieving a larger throughput via a more intense channel utilization. However, this strategy requires prolonged channel usage, which would be unfair in multiuser networks with random access techniques [6].

If full-duplex communications can be achieved by means of, e.g., time-division or frequency-division duplexing (TDD and FDD, respectively), more effective techniques based on Selective Repeat (SR)-ARQ can be employed. With SR-ARQ a window of up to $M$ consecutive packets can be transmitted before the sender stops to receive an ACK; moreover, retransmissions are limited to erroneous packets, at the price of a re-scheduling buffer at the receiver to cope with out-of-order receptions.[1]

To realize ARQ schemes based on SR techniques, a natural approach is to obtain a time-division duplex channel by leveraging on the propagation delays experienced by underwater sound. In typical shallow water network scenarios [7], the distance between neighboring nodes can be of the order of one

Manuscript received August 29, 2012; revised February 28, 2013; revised May 27, 2013; accepted June 20, 2013. The associate editor coordinating the review of this paper and approving it for publication was L. Cai.

Part of this work has been presented at the IEEE/OES OCEANS 2011 Conference, Santander, Spain.

This work was supported in part by Johns Hopkins University, Applied Physics Laboratory's internal research and development funds, by the European Commission under the Seventh Framework Programme (Grant Agreement 258359 – CLAM), and by the US National Science Foundation under Grant no. CPS-1035828.

S. Azad is with the Department of Computer Science, American International University–Bangladesh, Kemal Ataturk Avenue, Banani, Dhaka-1213, Bangladesh (e-mail: sazadm684@aiub.edu).

P. Casari and M. Zorzi are with the Department of Information Engineering, University of Padova, Via Gradenigo 6/B, 35131 Padova, Italy, and with Consorzio Ferrara Ricerche, via Saragat 1, 44122 Ferrara, Italy (e-mail: {paolo.casari, michele.zorzi}@dei.unipd.it).

Digital Object Identifier 10.1109/TWC.2013.121306.

[1]The Go-Back-N (GBN) ARQ technique [3] was also introduced in terrestrial networks to avoid the acknowledgement of out-of-order receptions, and thus save the storage space required for a re-sequencing buffer at the receiver. GBN is known to be outperformed by SR whenever the re-sequencing buffer can be afforded [3], hence it will not be considered here. In any event, we note that such buffer is well within the capabilities of any existing hardware to date.

or more kilometers, which turns into propagation delays of 1 s to 2 s.[2] The packet transmission times obtained with typical modem hardware in many underwater scenarios are shorter than these values [11], therefore employing S&W would turn into very low throughput.

In this paper, we jointly address the above issues by proposing and discussing Underwater Selective Repeat (USR), a SR-ARQ scheme which employs a form of TDD, and works well in combination with MAC protocols based on random access. The idea behind USR is that the typically long UW propagation delays should be exploited to pack several data packet transmissions within the same RTT, while keeping the receiver silent when it is expected to receive ACKs. This requires the transmitter to estimate the RTT. Note that, in the following, we define the RTT as the delay between the end of a data packet transmission and the beginning of the reception of the corresponding ACK, with reference to the case where both the data and the ACK transmissions are successful.[3] An estimate of the RTT can be obtained by measuring the timing of the data/ACK exchange. If the source has never transmitted packets to the destination before, this estimate is not available. This situation can be solved by performing a preliminary S&W exchange. If the RTT is sufficiently large, further packets for the same receiver (if any) can be sent sequentially: however, the timing of subsequent data transmissions must be adjusted so that the transmitter is never deaf to incoming ACKs. Note that the transmission of data packets and the reception of ACKs are interlaced in time, thereby avoiding the need to split data communications and feedback over separate channels.

While the RTT can be easily estimated with a preliminary S&W cycle as detailed above, other design choices are required, e.g., how many packets should fit within one RTT. In fact, there is a tradeoff between a higher point-to-point throughput if transmissions are packed tightly, and a better capability to avoid multiple-access interference when transmissions are separated by longer waiting times. This tradeoff will be detailed in Section III-D.

The use of some form of TDD to support multiple packet transmissions within a single RTT has also been considered in [12]–[14]. In [12], the authors propose to share the RTT equally between two communicating nodes, so that they may transfer an equal number of packets. However, this technique requires perfect time synchronization between the nodes, something that may be difficult to achieve even in static networks, and hinders the extension of the technique to multiuser networks. Another TDD-like technique, named Juggling-like-Stop-and-Wait (JSW), is proposed in [13]. With JSW, the sender transmits a fixed number of data packets as specified by a pre-calculated window size, and then waits for the related ACK/NACK before performing further transmissions. This does not allow a node to fall back to S&W cycles in the presence of low traffic (e.g., when a node has only one packet to transmit). Moreover, if the network is randomly deployed, it is suboptimal to choose the same window size for all nodes, as the RTT is generally different for different pairs of nodes. Finally, the protocol in [14] is designed to reduce the packet delivery delay over a single link, not in multiuser scenarios.

Unlike the approaches above, the scheme we propose in this paper adapts to the distance between any two communicating nodes, hence to the value of the RTT over the respective link. This adaptation does not have the objective to optimize the transfer of data over a single link: instead, it seeks the optimization of the network performance as a whole. Such a result is achieved by avoiding to pack as many data packet transmissions as possible within the same RTT, and by choosing transmission timings properly. Moreover, our proposed protocol does not require any time synchronization, and can be applied to any randomly deployed network (unlike, e.g., [12]) with any number of nodes (unlike, e.g., [13]). An option to make it robust in the presence of mobility is described in Section III-C.

## II. Underwater Selective Repeat (USR)

The USR protocol has been designed according to two guidelines: $i$) the underwater propagation delays are typically long with respect to the packet transmission time, and should be exploited for enabling a SR-ARQ scheme; $ii$) the scheme should be designed in a way that makes it suitable to multiuser networks.

USR works as follows. Assume that a source node generates packets for a given destination. The first time it makes contact with that destination, the source sends one packet using a common S&W scheme. Namely, it performs the channel access procedures required by the Medium Access Control (MAC) scheme in use, it sends one data packet, and waits for the corresponding ACK. If no ACK is received for this first data packet, the node will back off and another first-contact S&W cycle will take place at a later time. When the ACK is finally received, the node estimates the RTT between itself and the destination. If this time is too short to allow the transmission of multiple packets within one RTT, while still ensuring that data packet transmissions and ACK receptions will not collide, the transmitter falls back to S&W. The knowledge of the RTT allows the transmitter to estimate the length of the transmit window $M$, i.e., the maximum number of packets that can be transmitted before waiting for ACKs.

With reference to Fig. 1, call $\tau$ the propagation delay, and $T_D$ and $T_A$ the transmission time of a data packet and of an ACK packet, respectively. For a given destination,[4] the window size $M$ can be computed as

$$M = \max\left(1, \left\lfloor \frac{k\tau}{T_D + T_A + \delta} \right\rfloor\right) \tag{1}$$

where $\delta$ is a guard time which prevents tight scheduling of data packets and ACKs, and $k$ is an adaptation factor which limits the portion of the RTT to be considered in the computation of $M$, $0 < k \leq 2$. In the limit, if $k = 2$, the packets will be transmitted with the minimum possible spacing, whereas

[2]Many modems available to date support transmissions over such distances. For instance, AquaComm supports ranges of 3 km [8], the LinkQuest UWM2000 and UWM3000 modems support 1.5 km and 3 km, respectively [9], and the Evologics S2C R 48/78 modem supports 3.2 km [10].

[3]We remark that this definition is slightly different from what is used in some other contexts, where the RTT is defined as the time interval from the *beginning* of a data packet transmission to the *end* of the reception of the corresponding ACK.

[4]See Section II-C for details on the case of multiple destinations.

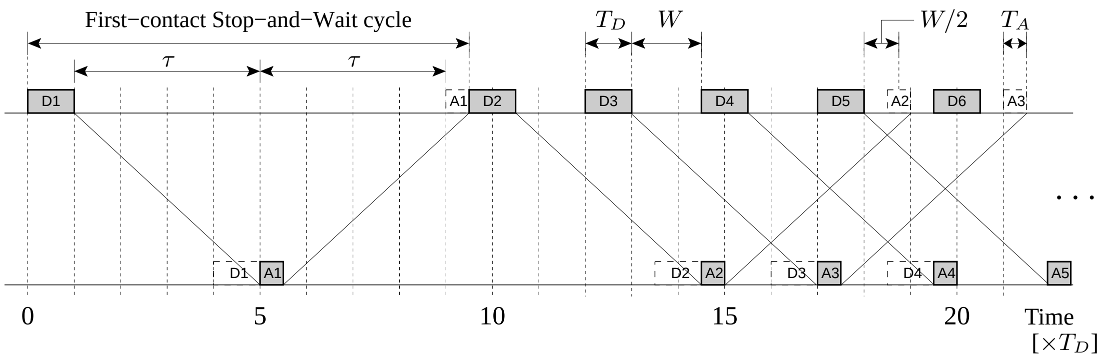

Fig. 1. Example of Underwater Selective Repeat (USR) in operation, for a transmit window $M = 4$. After the first contact, via a S&W cycle, $M$ and the waiting time W are computed from the RTT measurement. Relevant timings are highlighted. ($\tau = 4\,T_D$, $\delta = 0.5\,T_D$, $W = 1.5\,T_D$, $T_A = 0.5\,T_D$.)

if $k = 0$ the window $M$ will be lower-bounded by 1, which corresponds to falling back to S&W. The same happens if the communicating nodes are very close: in this case, the expression $\lfloor \frac{k\tau}{T_D+T_A+\delta} \rfloor$ in (1) is rounded to 0, and $M$ would be set to 1 as per the definition in (1). Note that, in a multiuser network, the RTT between different pairs of nodes may be different, and in general this reflects on different values of the window size $M$. The estimate of the RTT performed during the first-contact S&W cycle makes USR adapt to these differences. After that, the nodes continuously estimate any changes in the RTT (hence in $\tau$) by measuring the timing of all data-ACK exchanges, and by updating (1) accordingly. This makes it possible to keep the value of $M$ up to date, in the presence of RTT changes induced by mobility or by fluctuations in the physical parameters of the water. We note that, in static networks, the propagation speed typically changes negligibly over time, and is unlikely to spoil USR's interlaced DATA/ACK transmissions. Therefore, USR never falls back to S&W again, unless of course $M = 1$ from the computation in (1). The mobile network case requires special care and will be dealt with in Section II-B.

Whenever $M > 1$, the sender waits for a fixed time $W$ before sending the next packet of the window, in order to avoid receiving an ACK while transmitting a data packet. In the following, we compute $W$ in such a way that the ACK reception takes place in the middle of the waiting time, i.e., it is centered $W/2$ after the end of the reception of the previous data packet, if the RTT is constant. The waiting time $W$ is defined by the following relationship

$$T_D + \frac{T_A}{2} + 2\tau = MT_D + (M-1)W + \frac{W}{2}\,, \qquad (2)$$

where the left-hand side models the time that elapses between the beginning of a data packet transmission and the middle of the reception of the corresponding ACK, whereas the right-hand side stands for the fact that this reception should take place in the middle of the corresponding waiting time. By solving for $W$ we get

$$W = \frac{T_A + 4\tau - 2(M-1)T_D}{2M-1}\,. \qquad (3)$$

We remark that $W$ depends on $M$, which in turn is chosen such that there is always at least one data packet and one ACK transmission within a time interval of duration $k\tau$, $0 < k \le 2$. Since $W$ is derived from $M$, a smaller value of $M$ always results in longer waiting times between subsequent data packet transmissions. In any event, we observe that by imposing $W \ge T_A$ in (3) (i.e., the waiting time $W$ is always sufficiently large to accommodate an ACK), we get $M - 1 \le 2\tau/(T_A + T_D)$, which is always verified as per the expression in (1), meaning that $M-1$ data packets and ACKs always fit within one RTT.

By virtue of (2), the ACK related to a certain packet is expected $2\tau$ after the packet transmission, and around the middle of a time window of length $W$. Hence, $W$ in (3) also represents the timeout for the ACK reception. If the ACK is missing, for any reason,[5] the sender refrains from further transmissions using a standard binary exponential backoff scheme. The window of the backoff time is doubled at every failed ACK reception, and reset at the first successful one. After the backoff timer expires, a node always performs a fresh channel access attempt before transmitting again.[6]

In the following, we consider two versions of USR. In USR-SLiding Window (USR-SLW), i.e., the version reproduced in Fig. 1, a sender keeps sending packets to a given destination until an error occurs, or until all packets in queue for that destination have been transmitted. In USR-FiXed Window (USR-FXW), the sender transmits $M$ packets and then waits until all the corresponding ACKs have been received; only after that does it perform a further transmission of a window of $M$ packets. We finally note that the different behavior of the two USR versions requires different channel access patterns: in particular, the USR-FXW technique performs the access procedure before transmitting every window of $M$ packets; on the contrary, USR-SLW accesses the channel only before transmitting the first packet to a given destination. For this reason, USR-SLW strikes a different balance between how many transmissions are pushed towards the receiver per unit time and how much interference is generated in the area around the transmitter and the receiver.

The next subsections are organized according to the following roadmap. Section II-A provides analytical expressions for the throughput of USR-SLW and USR-FXW in a simple two-node scenario. The results are supported by Monte-Carlo

[5]We remark that ACKs lost due to collisions are not discriminated from ACKs lost because of channel impairments, and both events lead to backoff. This allows a node to escape either congestion or bad channel realizations. In particular, the latter show a quasi-periodic behavior in many scenarios [15], which can be escaped by refraining from transmitting for a sufficiently long time.

[6]These choices proved to offer the best performance in all simulations with multiple access interference. The main reason is that transmission patterns are not aggressive, and no two nodes seize the channel for themselves for long periods of time.

simulations, and provide first insight into the performance of the two policies. Sections II-B and II-C respectively describe how USR copes with node mobility, and provide further remarks on the choice of the transmit window $M$. More complex scenarios involving more than two nodes are discussed starting from Section III.

## *A. Throughput analysis*

Close-form expressions for the average throughput of USR-SLW and USR-FXW can be obtained by modeling the protocol operations as a renewal process. In USR-SLW, the renewal event is a packet error, due either to a failed data transmission or to the loss of the corresponding ACK. In USR-FXW, the renewal event is either a packet error or the correct transmission of $M$ consecutive data packets.

Focus on a point-to-point link between a source and a destination, and assume that the source is always backlogged.[7] Define a data packet transmission as successful if both the data packet and the corresponding ACK are correctly delivered. Therefore, the probability of success is $p = p_d\, p_a$ where $p_d$ is the probability that a data packet is correctly transmitted to its destination, and $p_a$ is the probability that the corresponding ACK is successfully received. Call $D$ the total time spent for data packet transmissions, and call $T$ the total length of a renewal cycle. For both USR versions, the average throughput $\bar{\theta}$ can be obtained as the ratio $\bar{\theta} = \mathbb{E}[D]/\mathbb{E}[T]$. The probability of transmitting exactly $i$ consecutive data packets successfully follows a geometric distribution:

$$P[i] = p^i(1-p)\,. \tag{4}$$

For USR-SLW, $\mathbb{E}[D_{\mathrm{SLW}}]$ is readily found as

$$\mathbb{E}[D_{\mathrm{SLW}}] = \eta\, T_D \sum_{i=0}^{+\infty} i\, p^i(1-p) = \frac{\eta\, T_D\, p}{1-p}\,, \tag{5}$$

where $\eta$ is the ratio of the length of the useful data in a data packet to the total packet length, and $T_D$ is the total data packet transmission time. In (5) and in the equations that follow we used the fact that

$$\sum_{i=0}^{j-1} i p^i(1-p) = \frac{p(1-p^j) - j p^j (1-p)}{1-p} \overset{j\to+\infty}{\longrightarrow} \frac{p}{1-p} \tag{6}$$

We recall that in every USR version, a data packet transmission is always followed by a waiting time $W$. Assuming that $i$ packets are successfully transmitted, whereas the $(i+1)$th is erroneous, the error will be detected via a missing ACK in the $M$th waiting time after the $i$th packet. After that, a backoff event of average duration $B$ always occurs. Therefore, the total duration of the renewal cycle, given $i$ consecutive successful data transmissions, is $T_{\mathrm{SLW}} = (i+M)(T_D+W)+B$, and its average is

$$\begin{aligned}\mathbb{E}[T_{\mathrm{SLW}}] &= \sum_{i=0}^{+\infty} i(T_D+W)\, p^i(1-p) + M(T_D+W) + B \\ &= \frac{(T_D+W)\big(M(1-p)+p\big) + B(1-p)}{1-p}\,,\end{aligned} \tag{7}$$

[7] If this is not the case, it would be unlikely to have enough packets to fill the transmit window $M$, hence USR-SLW and USR-FXW would achieve the same throughput, in line with the network simulation results shown in Section III.

where $M > 1$. The ratio of (5) on (7) yields USR-SLW's average throughput:

$$\begin{aligned}\bar{\theta}_{\mathrm{SLW}}(p) &= \mathbb{E}[D_{\mathrm{SLW}}]/\mathbb{E}[T_{\mathrm{SLW}}] \\ &= \frac{\eta\, T_D\, p}{(T_D+W)\big(M(1-p)+p\big) + B(1-p)}\,.\end{aligned} \tag{8}$$

In USR-FXW, at most $M$ consecutive packets are transmitted, after which the sender stops, in order to receive all ACKs. Assuming $M > 1$, the probability that $i$ data packet transmissions are successful is therefore

$$P[i] = \begin{cases} p^i(1-p), & i < M \\ p^M, & i = M \end{cases} \tag{9}$$

The average time spent for transmitting useful data is hence

$$\begin{aligned}\mathbb{E}[D_{\mathrm{FXW}}] &= \eta\, T_D \left[\sum_{i=0}^{M-1} i\, p^i(1-p) + M p^M\right] \\ &= \eta\, T_D\, p\, \frac{1-p^M}{1-p}\,.\end{aligned} \tag{10}$$

Following the same argument used for USR-SLW, in USR-FXW an erroneous transmission is detected because of a missing ACK during the $M$th waiting time after the transmission. However, USR-FXW transmits at most $M$ packets, after which it always waits for all ACKs to be received. Recalling that in case of errors a backoff of average length $B$ occurs, the total time spent before resuming transmissions is found as

$$T_{\mathrm{FXW}} = \begin{cases} (i+M)(T_D+W)+B, & 0 \le i \le M-1 \\ (2M-1)(T_D+W), & i = M \end{cases} \tag{11}$$

where $0 \le i \le M-1$ models the fact that an error occurs, whereas $i = M$ is the case where a complete window of $M$ packets is transmitted with no error, hence no backoff occurs. We have

$$\begin{aligned}\mathbb{E}[T_{\mathrm{FXW}}] &= (T_D+W)\left(\sum_{i=0}^{M-1}(i+M)p^i(1-p)(2M-1)p^M\right) \\ &\quad + B(1-p^M) \\ &= \frac{T_D+W}{1-p}\left(p + M(1-p) - p^M\right) + B(1-p^M)\,.\end{aligned} \tag{12}$$

The average throughput of USR-FXW is hence found as

$$\begin{aligned}\bar{\theta}_{\mathrm{FXW}}(p) &= \mathbb{E}[D_{\mathrm{FXW}}]/\mathbb{E}[T_{\mathrm{FXW}}] \\ &= \frac{\eta\, T_D\, p(1-p^M)}{(T_D+W)\,(p+M(1-p)-p^M) + B(1-p^M)(1-p)}\,.\end{aligned} \tag{13}$$

Note that $\bar{\theta}_{\mathrm{FXW}}(p)$ is undefined for $p = 1$. However, it can be continuously extended at $p = 1$ since

$$\lim_{p\to 1} \bar{\theta}_{\mathrm{FXW}}(p) = \frac{\eta\, T_D\, M}{(T_D+W)(2M-1)}\,, \tag{14}$$

which is in fact equal to the ratio of the time spent for sending useful data to the duration of a transmission cycle when all packet transmissions are successful. We finally remark that, for $M = 1$, both USR-SLW and USR-FXW fall back to a

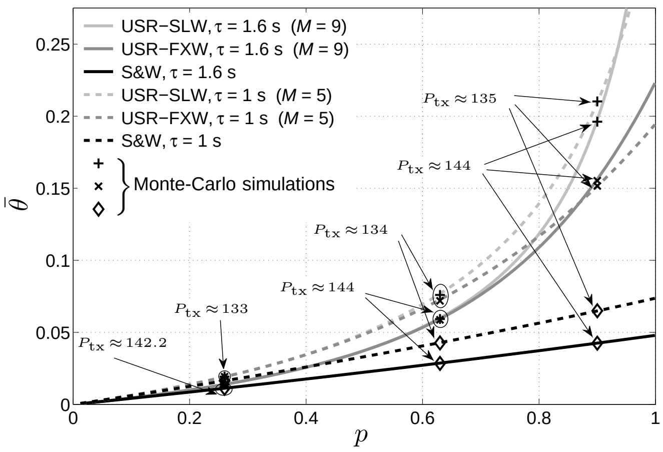

Fig. 2. $\bar{\theta}$ vs. $p$ for S&W and both versions of USR. Two different values of $\tau$ are considered, leading to different window sizes $M$. A larger value of $M$ makes USR achieve up to four times the throughput of S&W for sufficiently high $p$. $T_D = 220$ ms, $T_A = 18$ ms, $p_a = 1$, $\delta = 100$ ms, $k = 2$, $B = 500$ ms and $\eta = 0.75$. Monte-Carlo simulations employing the empirical link budget equations in [16] are shown to match the analysis well. All values of $P_{\rm tx}$ are expressed in dB re $\mu Pa^2$.

simple S&W protocol, whose average throughput is readily found as

$$\bar{\theta}_{\rm S\&W} = \frac{\eta\, T_D\, p}{T_D + 2\tau + T_A + B(1-p)} \,. \qquad (15)$$

Fig. 2 shows the throughput achieved by the USR and the S&W schemes in the presence of different values of the window size $M$, which in turn depends on the one-way propagation delay $\tau$ between the source and the destination. In this figure, we fixed $T_D = 220$ ms, $T_A = 18$ ms, $p_a = 1$, $\delta = 100$ ms, $k = 2$, $B = 500$ ms and $\eta = 0.75$, which represent realistic values for the system parameters and are akin to those found in the multiuser scenario in Section III. The results confirm the intuition that interlacing ACKs and data packets by properly timing the data packet transmissions yields higher throughput, compared to a plain S&W ARQ scheme. In particular, USR-SLW shows a consistently higher throughput than USR-FXW (which is expected, because the transmit window of USR-FXW does not slide), and both versions perform better than S&W, up to eight times for $p \approx 1$ and $M = 9$. We remark that a larger $M$ means higher throughput only if $p$ is sufficiently high. For instance, in Fig. 2 USR-SLW with $M = 9$ outperforms the case with $M = 5$ only if $p > 0.92$. The reason is the higher value of $\tau$ in the $M = 9$ case, hence the longer time that must elapse after an error for all pending ACKs to be received.

The analytical results in Fig. 2 are supported by Monte-Carlo simulations, in which realistic values have been set for the parameters of the communications system. In particular, we assume that the nodes employ a 4500-bps Binary Phase-Shift Keying (BPSK) modulation scheme with a center frequency of 25 kHz and a system bandwidth of 9 kHz. The data packet and the ACK packet sizes are fixed to $L_D = 125$ Bytes and $L_A = 10$ Bytes, respectively. Taking a representative value of 1500 m/s for the sound speed, the transmitter and the receiver are placed at a distance of 2400 m in the case $\tau = 1.6$ s, and at a distance of 1000 m in the case $\tau = 1$ s. The attenuation incurred by the acoustic signal over such distances is computed via the empirical link budget equations presented in [17], see also [16, Eqs. (1)–(4)]. The noise power is computed at the center frequency and assumed white over the system bandwidth. Moderate shipping and no wind conditions are assumed in order to compute noise ($S = 0.5$ and $w = 0$ in [16, Eq. (6)]). The transmit source level $P_{\rm tx}$ is varied to tune the probabilities $p_d$ and $p_a$, which are derived using the BPSK bit error equations and assuming independent bit errors over a packet. The chosen values are reported in Fig. 2 for each set of simulation points. Finally the probability $p = p_d\, p_a$ is computed to match the simulations to the analytical curves in Fig. 2. In all cases, the simulations are shown to match the analysis very well. We stress that the Monte-Carlo simulations include the first-contact S&W cycle, whereas the analysis neglects it: the good match between analysis and simulation confirms that such S&W cycle is negligible in the long run.

## B. Accounting for mobility in USR

Normally, the setting of $M$ and $W$ in (1) and (3), respectively, guarantees that the transmitter is never deaf to ACKs from the receiver. However, if the transmitter and the receiver are mobile, the RTT may vary over time, and possibly lead to the superposition of data transmissions and ACK receptions. This effect must be compensated both $i$) during a transmission phase and $ii$) across subsequent transmission phases. For case $i$), at each ACK packet received, the data sender re-estimates the RTT via the timing of the data-ACK exchange. The new estimate is employed to update the current values of $M$ and $W$. For case $ii$), in order to compensate for the RTT differences between subsequent transmission phases, we take a conservative approach and assume that the two nodes moved towards each other in the meantime. This has the net effect of yielding a lower value of $M$ (and thereby larger values of $W$) than in the static case. In more detail, assume that node A transmits data to node B using USR, and call $\tau_{AB}$ A's estimate of the RTT resulting from the last transmission phase between A and B. Assume that A and B meet again after a time $\Delta T$. A will compute $M$ by employing the propagation delay estimate $\tau'_{AB} = d'_{AB}/c$, where $c$ is the speed of sound (approximated using the fixed value of 1500 m/s for the purposes of this computation), and

$$d'_{AB} = \max\left(d_{AB} - \Delta T\, v\,,\, 0\right) \qquad (16)$$

is computed by assuming that the nodes move towards each other. In (16), $v$ is the maximum relative velocity of the nodes, computed as the sum of the speeds of A and B (B can make A aware of its speed by piggy-backing the corresponding value in the ACK packets). Unlike in the static case, here $d'_{AB}$ is a worst-case estimate of the distance between the moving nodes, and hence leads to a lower bound on the propagation delay. We remark that the nodes need not know their position or direction of movement.

## C. Remarks on $M$

It is important to note that the transmit window size $M$ is directly related to the destination, and is affected by the distance between the communicating nodes and by their relative movement pattern. Therefore, in general, a different value

$M_{\ell n}$ must be maintained and updated for each destination $n$ a node $\ell$ may transmit to.

Assume that a sender A holds packets for different destinations in its buffer, say B, C and D. Assume the first packet in A's queue is for node D. A would search the queue for packets with destination D, which are not contiguous in the buffer in general, and start a transmission towards D using the USR scheme with a window value equal to $M_{AD}$, possibly modified as per the instructions in Section II-B in case of mobility. During the transmission procedure, $M_{AD}$ is continuously updated via the repeated evaluation of the RTT between A and D, measured from the timing of data-ACK exchanges. When the transmission ends (because all packets have been transmitted or because an ACK is lost), the node backs off. At the end of the backoff period, node A checks the destination of the first packet in its queue again. In general such destination may be a different node, e.g., B. Hence the transmission procedure will take place using the appropriate transmit window $M_{AB}$.

## III. Simulation Results

This section presents the performance evaluation of the USR protocol in several scenarios. First, we provide a description of the scenario and of the system parameters in Section III-A; we illustrate the simulation results for static networks in Section III-B and proceed to the results related to mobile networks in Section III-C; in Section III-D, we show the impact of the parameter $k$ introduced in (1). Based on the considerations provided in the latter section, an adaptive version of USR is proposed in Section III-E.

### A. Scenario definition and common parameters

The USR protocol is designed as a complement to Medium Access Control (MAC) protocols. In this paper, we stack USR on top of a 1-persistent Carrier-Sense Multiple Access (CSMA) scheme tested in underwater networks in [18]. This instance of CSMA is described as follows: a node that has a packet to transmit senses the channel first. The sensing time is random, and very short with respect both to the transmission time of a data packet and to the maximum RTT in the network. The random length of the sensing time makes it possible to avoid the synchronization of channel access attempts performed by different nodes. If the channel is found busy, the node immediately performs a second sensing phase, again of random length, and reiterates this process until the channel is eventually sensed idle. At this point, the node transmits. We remark that channel sensing is only applied upon the first packet transmission that takes place during any contact between two nodes: ACK transmissions are not subject to channel sensing, and are regulated only by USR's timings as described in Section II. CSMA's preliminary carrier-sensing period makes it possible to avoid some typical collision events via a less aggressive access behavior (unlike what would happen, e.g., with ALOHA). In addition, CSMA has been found to provide good performance [18] with respect to a scheme based on collision avoidance [19] and to a contention-based scheme relying on wakeup tones [20]: this motivates its adoption for evaluating USR in this paper.

In this paper, the USR-CSMA pair will be compared against CSMA with a plain Stop&Wait (S&W) ARQ scheme, and against an ALOHA channel access protocol (whereby a node transmits a packet as soon as it is generated, unless the node is already engaged in another packet transmission), also coupled to S&W ARQ. For all protocols, the maximum number of retransmissions allowed for a packet is limited to 5. If the reception of an ACK fails for any reason, the transmitter resorts to exponential backoff. As to USR, the backoff window is doubled upon successive failed transmissions, and reset upon the reception of an ACK.

Unlike the Monte-Carlo simulations in Section II-A, the simulations we carry out in this section are performed using the ns2-Miracle framework [21]. This makes it possible to achieve a realistic reproduction of the node behavior in the presence of complex interactions related, e.g., to multiple channel access. In addition, we make use of the World Ocean Simulation System (WOSS) [18], which provides a channel model of improved accuracy, achieved through the Bellhop ray tracing software [22]. More specifically, WOSS retrieves the geographical coordinates of the nodes from ns2-Miracle and queries oceanographic databases for environmental data measured nearby the network deployment area. (In particular, a typical July SSP retrieved from the WOD [23] is employed throughout our simulation campaign.) This data is fed to Bellhop in order to generate channel realizations.

In all simulations, the network area is located in the Mediterranean Sea, near the Corsica Island, France. The upper left corner of this area is set at $(43.0625°N, 9.3095°E)$. the area extends over a square surface with side 2500 m, and over a maximum depth of about 80 m. All nodes are randomly deployed within the area. In static scenarios, the depth of all nodes is fixed to 80 m. In mobile scenarios, the depth of each node is randomly chosen at the beginning of each simulation run, and held constant for the whole run.

The system parameters are set as described in Section II-A. In addition, we remark that the transmission time of a data packet and of an ACK packet are $T_D \approx 220$ ms and $T_A \approx 18$ ms, respectively. For the computation of (1), we set the guard time $\delta = 100$ ms, i.e., about $T_D/2$. This proved to offer the best throughput in our simulations. The source level has been set to 200 dB re $\mu Pa^2$, making the network fully connected with high probability. In turn, this helps put the error control protocols under stress. We note that the probability of error over each link may still vary, due to the different channel realizations obtained from Bellhop. This scenario is reasonable for a network where other upper-layer protocols (e.g., routing) and applications (e.g., monitoring, remote control, telemetry) generate communication flows which need to be error-resilient. In turn, such communication flows may cross or interact in a number of fashions, and thereby interfere. Our scenarios allow us to abstract from the behavior of upper-layer protocols, while still being able to model the effects of such cross-interference on the ARQ protocols under study and on the channel access schemes they are coupled with.

In every simulation run, each node generates packets according to a Poisson process of normalized rate $\lambda$ packets per packet transmission time. Every node randomly chooses a destination, and transmits all its packets to that destination

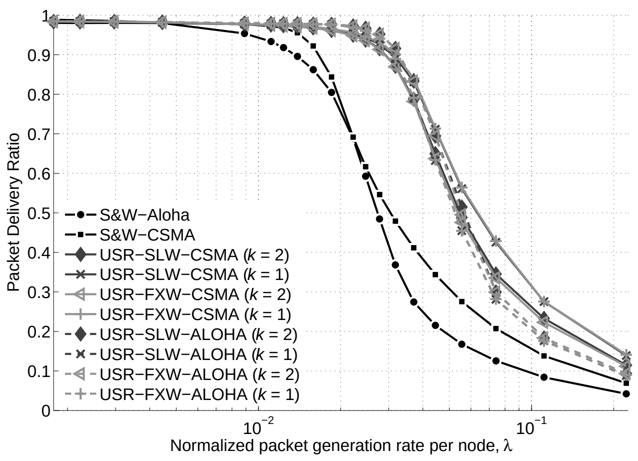

Fig. 3. PDR vs. $\lambda$ for all protocols in a static network of 5 nodes.

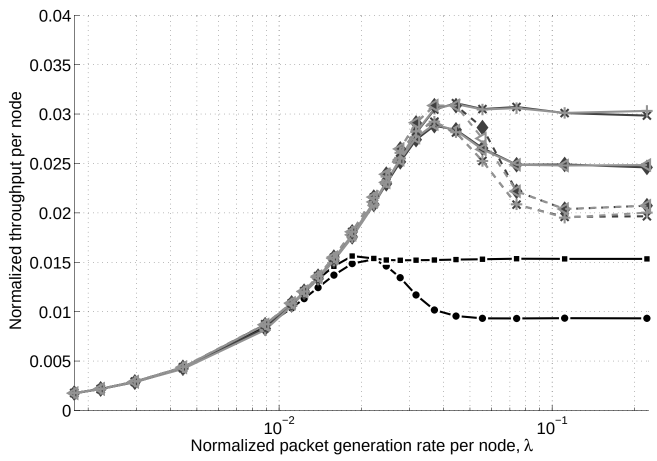

Fig. 4. Normalized throughput per node vs. $\lambda$ for all protocols in a static network of 5 nodes. (Key: see Fig. 3.)

throughout the whole simulation run. This means, for example, that a node A can choose node B as its destination, and be chosen as a destination by nodes C and D. Such associations are redrawn at the beginning of each run. All simulation results are averaged over a total of 25 simulation runs, which was found to yield sufficient statistical confidence. The simulations have been performed for several network area sizes, in both static and mobile networks, and for several values of $\lambda$. The latter include both the linear traffic and the saturation traffic regimes.

## B. Static network

We start by considering the case of a static network with a fixed number of nodes. We will refer to the protocols with a shorthand name that includes both the ARQ scheme and the channel access protocol in use: S&W-ALOHA and S&W-CSMA indicate the use of a S&W ARQ scheme on top of ALOHA and CSMA, respectively, whereas USR-FXW-CSMA, USR-FXW-ALOHA, USR-SLW-CSMA and USR-SLW-ALOHA denote the use of either version of USR along with the CSMA or ALOHA access scheme.

Figs. 3 and 4 show the average Packet Delivery Ratio (PDR) (defined as the ratio of the number of packets correctly received by their intended destinations to the number of generated packets[8]) and the normalized throughput per node (defined as the number of packets correctly delivered in the network per packet transmission time per node) in a static network of 5 nodes. All protocols perform similarly for small values of $\lambda$, corresponding to a light network load and a low probability of collision. As $\lambda$ increases, S&W-CSMA's channel sensing procedure prevents some collisions, hence the protocol achieves better PDR and throughput than ALOHA. In this scenario, the greatest throughput achieved by both S&W-ALOHA and S&W-CSMA is around 0.08. However, S&W-CSMA's sensing procedure makes it more robust in the face of heavy traffic, hence its throughput curve remains stable at a value close to its maximum, whereas S&W-ALOHA's decreases.

[8]All packets left in the buffer of a node at the end of a simulation are counted as lost packets.

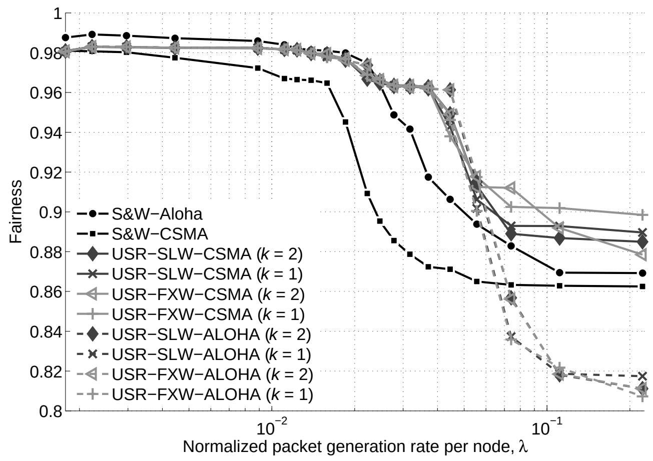

Fig. 5. Throughput fairness vs. $\lambda$ for all protocols in a static network of 5 nodes.

For all values of $\lambda$, both versions of USR on top of CSMA achieve a better PDR than the other protocol stacks. This is due both to the use of a TDD scheme for duplexing data packets and ACKs, and to the transmission of multiple packets within one RTT; altogether, these features translate into a higher normalized throughput per node. In Section III-D we will discuss the impact of $k$ in more detail: at this time, we simply note that $k = 1$ leads to better performance than $k = 2$. Intuitively, the reason is that a lower value of $k$ leaves longer silence periods between subsequent packets. In turn, this makes it less likely to experience collisions generated by hidden terminals,[9] or by the fact that transmitters are deaf to packets meant for them. On the contrary, $k = 2$ yields a higher number of transmissions per unit time. This would be optimal for a single link, but translates into greater interference in a multiuser network, originating more collisions and more backoff events. The transmission errors, and the silence periods that ensue, ultimately lead to a throughput loss. We note that the use of

[9]Hidden terminals tend to appear because of the large vulnerability time imposed by the underwater acoustic channel. In fact, the typical propagation speed of acoustic signals under water is on the order of 1500 m/s. The maximum distance in our static scenario is about 3.5 km, which leads to a maximum vulnerability time of about 2.3 s.

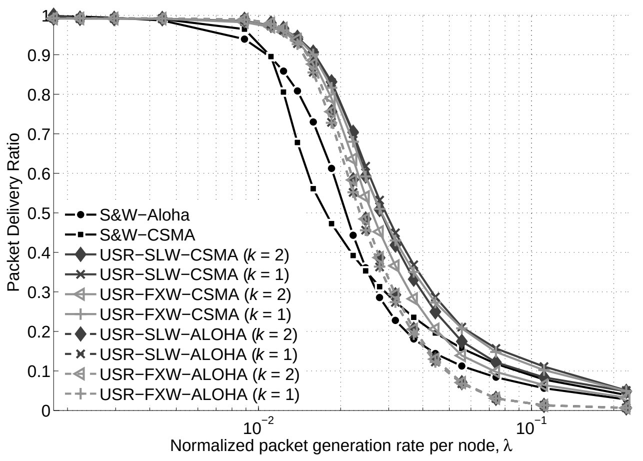

Fig. 6. PDR vs. $\lambda$ for all protocols in a static network of 10 nodes.

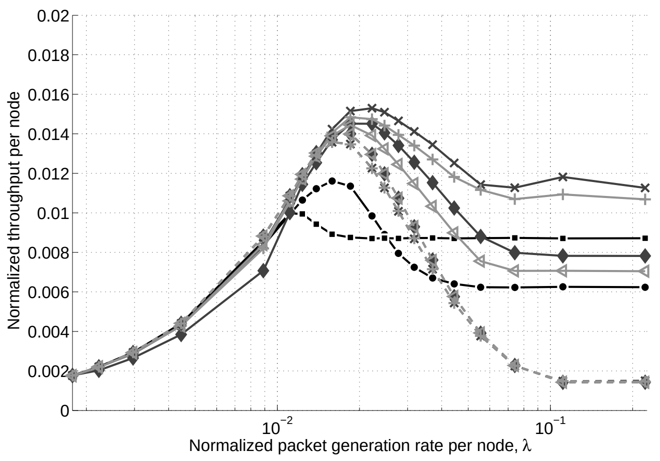

Fig. 7. Normalized throughput per node vs. $\lambda$ for all protocols in a static network of 10 nodes. (Key: see Fig. 6.)

ALOHA as a channel access scheme leads to a throughput loss caused by the more frequent occurrence of collision and deafness events. In any event, the loss is limited in Figs. 3 and 4 because of the limited number of network nodes.

Fig. 5 shows Jain's fairness index $\mathcal{F}$ as a function of $\lambda$. $\mathcal{F}$ is defined as

$$\mathcal{F} = \frac{\left(\sum_{n=1}^{N} \theta_n\right)^2}{N \sum_{n=1}^{N} \theta_n^2} \,, \tag{17}$$

where $N$ is the number of nodes in the network (5 in the present scenario) and $\theta_n$ is the throughput experienced by node $n$. We observe that all protocols achieve a fairness around $0.98$ for $\lambda < 10^{-2}$, which progressively decreases as $\lambda$ increases towards the saturation region. All CSMA-based USR versions perform better than S&W-based schemes, showing that for this network size the use of multiple packet transmissions within the same RTT does not lead to starvation. Notably, this achievement is a combined effect of the ARQ scheme and the channel access scheme: in fact, the USR versions stacked on top of ALOHA experience the lowest fairness among all schemes. This is mainly due to the fact that CSMA makes it evenly likely for all nodes to back off in the presence of other transmissions. On the contrary, the more aggressive channel access pattern enforced by ALOHA eventually makes some nodes prevail over other nodes that will experience more frequent backoffs.

Figs. 6 and 7 show the PDR and normalized throughput per node for all protocols in a network of 10 nodes. Because $\lambda$ is the packet generation rate per node, the presence of 10 nodes effectively doubles the network load with respect to the case of Figs. 3 and 4. The heavier contention that results highlights the difference between S&W-ALOHA (which performs better at low traffic) and S&W-CSMA (which outperforms S&W-ALOHA for $\lambda > 0.022$). The difference between the $k = 1$ and the $k = 2$ cases still remains, and is explained in the same way as in the 5-node scenario: $k = 1$ leaves longer periods of silence between subsequent data transmissions, hence reducing the probability of collisions, with beneficial effects on both the PDR and the throughput. With respect to Figs. 3 and 4, in Figs. 6 and 7 the use of USR stacked on top of ALOHA leads to larger performance losses, in terms of success ratio, peak throughput and saturation throughput.

Such losses are also observed from the fairness curves plotted in Fig. 8, where all USR versions reach a fairness of about $0.35$ when used along with ALOHA. On the contrary, USR with CSMA achieves a high fairness, comparable to that of the S&W-based schemes, which do not resort to multiple transmissions within the same RTT. This confirms that starvation is not a prominent issue in USR. The best fairness for USR is observed for the case $k = 1$, consistent with previous comparisons. With respect to USR, S&W-CSMA achieves a slightly higher fairness for $\lambda > 0.04$. However, this advantage is a symptom that all nodes experience a globally lower throughput, as confirmed by Fig. 7. We note that both versions of USR perform well: the throughput difference between USR-SLW-CSMA and USR-FXW-CSMA for high values of $\lambda$ in Fig. 7 is explained by recalling that the latter sends a train of packets and then waits for all their corresponding ACKs. This forces deterministic silence periods for all transmitters and leads to a lower throughput.

It is interesting to note that, for $k = 2$, both versions of USR achieve lower throughput than S&W-CSMA in the scenario with 10 nodes for $\lambda > 0.06$. This further supports our observation above, as S&W-CSMA employs a S&W ARQ scheme, which inherently gives rise to silence periods longer than those employed by USR. In other words, the transmission of multiple packets within the same RTT does not always result in good performance in multiuser networks: tuning USR's $k$ parameter as a function of the total network traffic is therefore key to making USR outperform S&W.

In Fig. 9, we consider a network of 5 nodes and increase the length of the side of the network area from 0.1 km to 3 km. When nodes are very close to each other, channel sensing gives benefit to S&W-CSMA and to both USR versions, when coupled with CSMA. However, when the area is small, the average RTT is lower, hence USR rarely has a chance to interlace data packet transmissions and ACK receptions; as a consequence, the performance improvement offered by USR is limited (if any). The opposite occurs for larger areas up to 3 km of side: in this case, the average distance between a sender and its receiver is higher, and the longer propagation delays that result are better exploited by USR than by S&W-

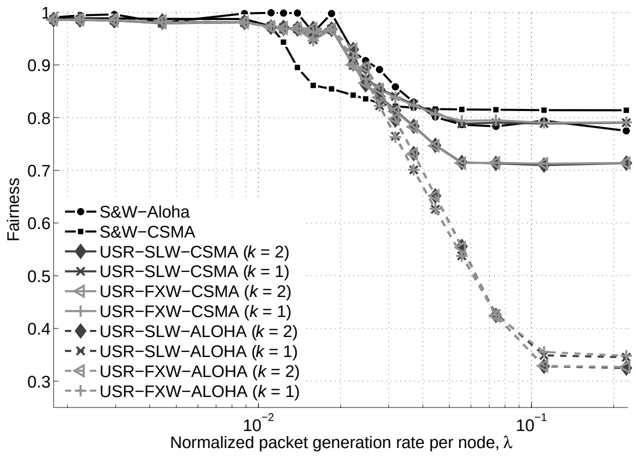

Fig. 8. Throughput fairness vs. $\lambda$ for all protocols in a static network of 10 nodes.

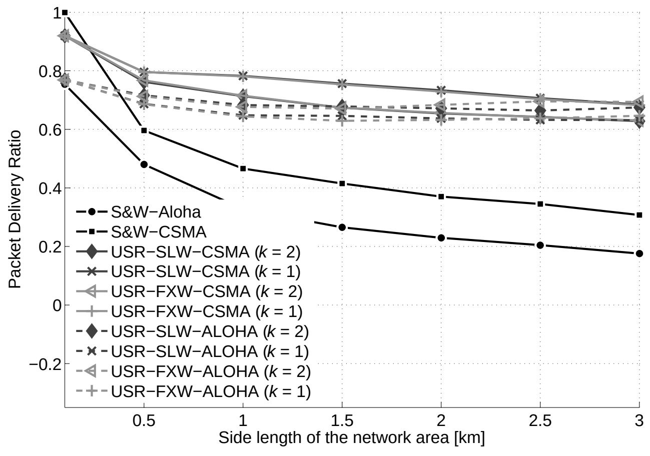

Fig. 9. PDR vs. the length of the side of the network area for all protocols, 5 nodes, $\lambda = 0.044$.

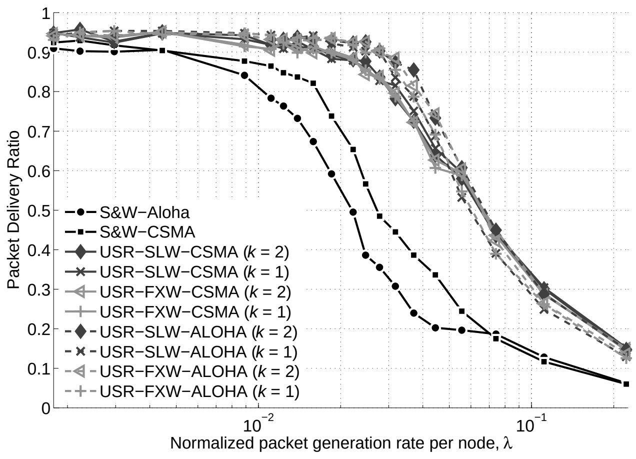

Fig. 10. PDR vs. $\lambda$ for all protocols in a mobile network of 5 nodes.

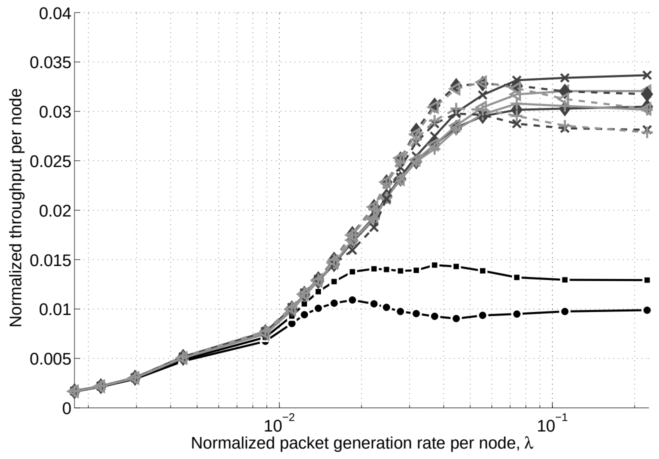

Fig. 11. Normalized throughput per node vs. $\lambda$ for all protocols in a mobile network of 5 nodes. (Key: see Fig. 10.)

ALOHA and S&W-CSMA. As a result, the PDR of the two USR versions is still between 0.6 and 0.7. In particular, USR-SLW-ALOHA and USR-FXW-ALOHA perform slightly worse than their CSMA counterparts, due to the increased number of collisions and deafness events. On the contrary, the PDR of S&W-ALOHA and S&W-CSMA drops to 0.3 or less. We finally note that the choice of $k = 1$ or $k = 2$ leads to similar differences as those observed in Figs. 3 and 6.

### C. Mobile network

We now consider a mobile network of 5 nodes. Each node starts from a random position and depth, and moves within the network area according to a Gauss-Markov mobility model [24] with fixed correlation parameter $0.8$. Hitting the boundaries of the network area causes the nodes to bounce back. The depth of each node is kept constant throughout each simulation run.

Figs. 10 and 11 respectively show the PDR and the normalized throughput per node for all protocols in a mobile network of 5 nodes. We recall that the maximum number of retransmissions is limited to 5: this limit is sometimes exceeded, leading to packet drops, and explaining why the delivery ratio of the protocols never reaches 100%. Note that there is little difference between the PDR and throughput of the different versions of USR. The dominating factor, in this case, is that USR always assumes that the transmitter and its receiver move towards each other between subsequent contacts, hence a fallback to S&W eventually occurs. Only thereafter do the nodes re-estimate the RTT (hence $M$ and $W$) using the timing of the data-ACK exchange, and resume the normal USR behavior. In any event, fallbacks to S&W in the presence of mobility are frequent for both USR-SLW and USR-FXW, reducing the number of transmissions injected in the network, and smothering the difference between the two USR versions. We observe that the highest throughput is achieved by USR-SLW-CSMA when $k = 1$, and is twice as high as that of S&W-CSMA. The limited traffic resulting from the S&W fallbacks also explains the better performance of ALOHA-based protocol stacks in the linear throughput region. When $\lambda > 0.1$, this tendency is inverted, as the aggressiveness of ALOHA generates more errors (see Fig. 10) and a progressively lower throughput as $\lambda$ increases (Fig. 11). In any event, the difference between USR over ALOHA and USR over CSMA is limited especially in the high throughput region, and leads to the conclusion that CSMA-based USR versions offer good performance in a broader range of scenarios.

We finally note that the PDR experienced by the protocols

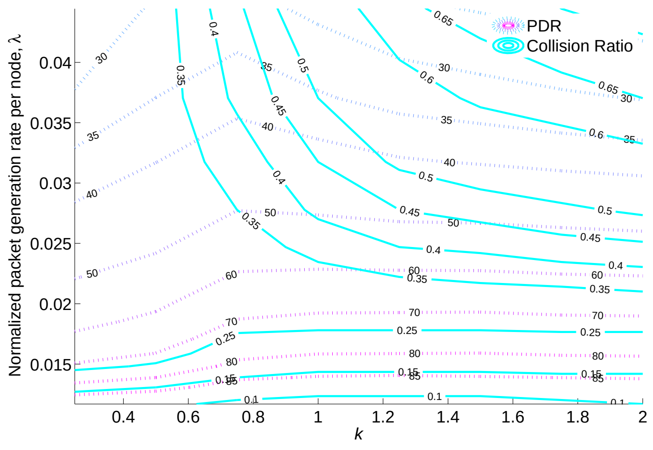

Fig. 12. Contour curves of the PDR and of the collision ratio for USR-FXW-CSMA in a static network of 10 nodes. Curves are plotted as a function of $k$ and $\lambda$. Each contour curve is obtained as the intersection of the PDR or collision ratio surfaces as a function of $k$ and $\lambda$ with a horizontal plane corresponding to the PDR or collision ratio value indicated by the label on each curve.

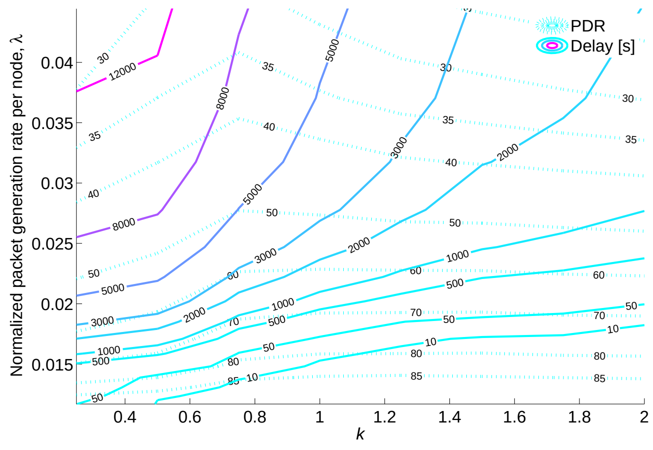

Fig. 13. Contour curves of the PDR and of the delivery delay for USR-FXW-CSMA in a static network of 10 nodes. Curves are plotted as a function of $k$ and $\lambda$. Each contour curve is obtained as the intersection of the PDR or delivery delay surfaces as a function of $k$ and $\lambda$ with a horizontal plane corresponding to the PDR or delay value indicated by the label on each curve.

in a mobile network is slightly lower than in a static network, as can be observed by comparing Fig. 10 to Fig. 3. This is due to the realistic channel realizations provided by the WOSS package, which vary over time in a mobile network, causing more frequent transmission errors.

## D. *The impact of $k$*

In the previous subsections, we have observed that $k$ can be used to adapt the behavior of USR: in particular, $k$ tunes the number of transmissions performed within one RTT, and therefore the timings these transmissions are subject to. We have intuitively explained that $k = 2$ is a good setting for the optimization of point-to-point scenarios, whereas any value $k < 2$ reduces the frequency of transmissions in time, and therefore leads to a lower probability of losing packets because of collisions. This suggests that $k < 2$ may be a good choice in multiuser networks. However, decreasing $k$ too much would be detrimental in terms of throughput, especially if the RTT between the transmitter and the receiver is very high.

In this section, we explain this tradeoff in more detail, and give some guidelines for the choice of $k$ as a function of the scenario parameters and of the metrics to be optimized. For consistency with Subsection III-E and because USR-SLW-CSMA and USR-FXW-CSMA lead to similar PDR and throughput in multiuser networks for a fixed value of $k$, we only consider USR-FXW-CSMA in this subsection. We focus on the case of a 10-node network: the conclusions for a 5-node network are entirely analogous.

We start with Fig. 12, which depicts the PDR and the collision ratio (i.e., the fraction of packets lost due to collisions) as a function of $k$ and $\lambda$. The dependence of these metrics on such parameters is shown by means of contour plots, where each curve is obtained as the intersection of the PDR and collision ratio surfaces with a horizontal plane corresponding to the value indicated by the label on the curve. First, we observe that for $\lambda < 0.022$, increasing $k$ improves the PDR by allowing the nodes to resort more often to large transmission windows ($M > 1$ in (1)); in turn, this reduces the chance that there are still packets in the queue of the nodes at the end of a simulation, and that these packets are counted as lost. Note that this does not increase the number of collisions, as the traffic generated by the nodes is still very low. For higher values of $\lambda$, increasing $k$ improves the PDR for the same reasons above, but only roughly until $k < 0.75$, after which the PDR starts decreasing for increasing $k$. The reason is the higher chance that two transmissions collide: in fact, the collision ratio also increases.

A similar analysis can be applied to the comparison of PDR and delivery delay, defined as the average time required to deliver a packet to its destination node. Such analysis is presented in Fig. 13. The figure shows that increasing $k$ always leads to a lower delay, as a consequence of the denser packing of packet transmissions within the same RTT. In turn, if $\lambda > 0.022$, this leads to a lower PDR. For a given value of $\lambda$, Fig. 13 helps choose the best value of $k$ that achieves the desired PDR and delay, and also highlights which values of these metrics are not achievable. For instance, $k = 1$ achieves a PDR of less than $60\%$ and a delay of about 2000 s for $\lambda \approx 0.022$. Increasing $k$ helps reduce the delay while maintaining the PDR almost constant. However, it is impossible to achieve, e.g., a PDR of 80% and a delay of less than 10 s for $\lambda = 0.022$. However, if $\lambda \approx 0.015$, such constraints can be jointly achieved for any $k \geq 1.16$.

## E. *USR Additive Increase–Multiplicative Decrease (USR-AIMD)*

The previous subsection explains that $k$ can be tuned in order to achieve a given set of constraints on the PDR, the collision ratio and the delivery delay. Which value should be chosen depends on, e.g., the amount of traffic generated per unit time in the network. As these conditions may be subject to changes over time (for example due to local congestion events), in this section we design an algorithm that adapts the value of $k$ depending on the capability of a transmitter to

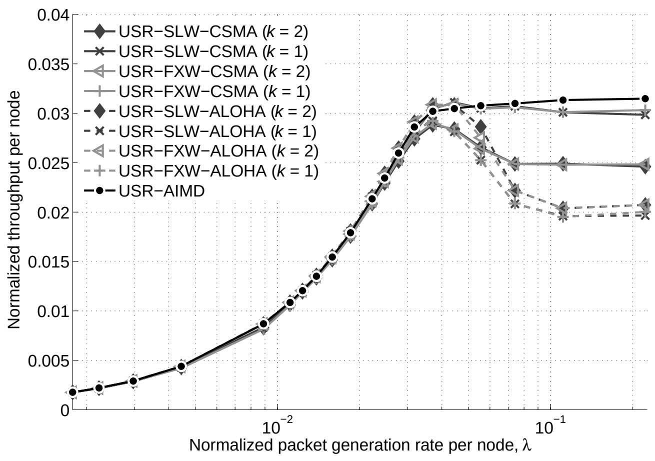

Fig. 14. Normalized throughput per node vs. $\lambda$ for all versions of USR in a static network of 5 nodes.

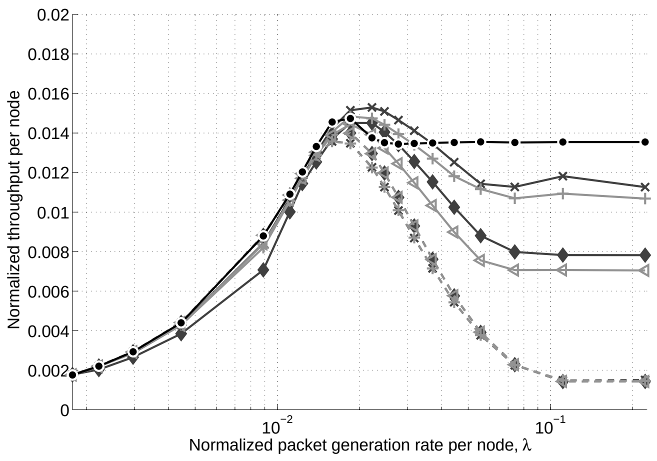

Fig. 15. Normalized throughput per node vs. $\lambda$ for all versions of USR in a static network of 10 nodes. (Key: see Fig. 14.)

deliver packets without errors. We name this technique USR-Additive Increase Multiplicative Decrease (USR-AIMD), as it is inspired by the well-known window adaptation mechanism of the Transport Control Protocol (TCP)'s congestion avoidance mode. USR-AIMD is implemented as an extension of USR-FXW (which sends a given number of packets within one RTT and then waits for all ACKs to be received). This way, the updates of the window length occur only after both the transmission of a window of packets and the reception of the corresponding ACKs have been completed (or, in case of errors, after the ACK timeout has fired).

As the name suggests, for each successful transmission, the window size $M$ increases until any packet loss occurs, and a retransmission must be performed. Specifically, focus on the link between a source A and its destination B. We increase the window size as follows

$$M' = \min\left(M_{AB}, M+1\right) , \tag{18}$$

where $M_{AB}$ is the maximum window size computed for A and B, and $M$ is the previous window size. If any packet is lost (the corresponding ACK is not received), the window is simply decreased according to the factor $0 \leq \alpha < 1$.

$$M' = \max\left(1, \lfloor \alpha M \rfloor\right) . \tag{19}$$

Figs. 14 and 15 show the normalized throughput per node achieved by all versions of USR, including USR-AIMD, in a network of 5 nodes and in a network of 10 nodes, respectively. We excluded S&W-ALOHA and S&W-CSMA from this comparison, since they are consistently outperformed by USR in terms of throughput. In both figures, USR-AIMD shows the same performance as the other USR versions at low traffic, and outperforms them when the traffic is sufficiently high ($\lambda > 0.07$ in the 5-node network and $\lambda > 0.04$ in the 10-node network). We notice that in the intermediate traffic regime, USR-AIMD is outperformed by the other versions. This is due to the multiplicative decrease of the window length, which makes USR-AIMD refrain from performing too many data transmissions, in case some packets do not get through. The same feature also allows USR-AIMD to keep the throughput stable at high traffic, reaching a larger value than that achieved by the other versions of USR.

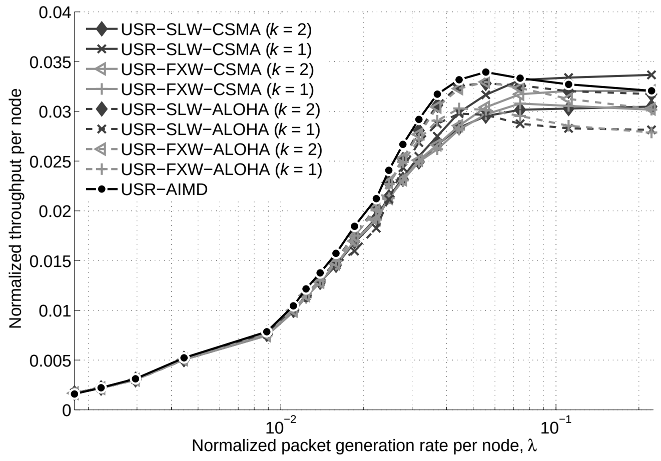

Fig. 16. Normalized throughput per node vs. $\lambda$ for all versions of USR in a mobile network of 5 nodes.

As a final remark, USR-AIMD helps improve the throughput performance in mobile networks as well. Consider for instance the scenario with 5 nodes discussed in Section III-C. Fig. 16 reproduces the USR curves in Fig. 11 (excluding S&W-based stacks) and compares them to the performance of USR-AIMD. The latter achieves the best throughput among all USR versions in the linear throughput region. This advantage is due to the adaptation of the transmit window both to the distance between the nodes and to the traffic pattern, which are very time- and location-dependent in a mobile network. The multiplicative decrease behavior, in addition to the conservative rules of Section II-B, makes the throughput decrease slightly for $\lambda > 0.06$, as it achieves a value in line with the $k = 1$ versions of USR.

## IV. CONCLUSIONS

In this paper, we proposed Underwater Selective Repeat (USR), an Automatic Repeat reQuest (ARQ) scheme for multiuser underwater acoustic networks. The scheme relies on time division in order to set up a duplex channel between the transmitter and the receiver, so that the transmission of data packets can be interlaced with the reception of the corresponding acknowledgments (ACKs). We evaluated the performance of USR by means of simulation in static and mobile networks, and showed that USR outperforms other protocol stacks relying on plain Stop&Wait. We then observed that by limiting the ARQ window length (i.e., the number of data packet transmissions that can be performed before receiving an ACK) via the parameter $k$, the network achieves better delivery ratio and throughput; in addition, we commented on the relationship between the traffic generation rate, the factor $k$ and such network metrics as the packet delivery ratio and the delivery delay. In particular, we showed how $k$ should be tuned in order to achieve a given set of constraints on these metrics. We finally designed USR-AIMD, an adaptive version of USR which automatically adapts the window length, consistently improves the throughput in both static and mobile networks at low to intermediate traffic, and achieves a stable throughput at high traffic. We endorse the latter as a good candidate for implementation in real multiuser underwater networks.

## References

[1] A. Valera, P. W. Q. Lee, H.-P. Tan, H. Liang, and W. K. G. Seah, “Implementation and evaluation of multihop ARQ for reliable communications in underwater acoustic networks,” in *Proc. IEEE/OES Oceans*, Bremen, Germany, May 2009.

[2] R. K. Creber, J. A. Rice, P. A. Baxley, and C. L. Fletcher, “Performance of undersea acoustic networking using RTS/CTS handshaking and ARQ retransmission,” in *Proc. MTS/IEEE Oceans*, Honolulu, HI, USA, Nov. 2011.

[3] B. A. Forouzan and S. C. Fegan, *Data Communications and Networking*. McGraw-Hill, 2003.

[4] M. Stojanovic, “Optimization of a data link protocol for an underwater acoustic channel,” in *Proc. IEEE Oceans*, Brest, France, June 2005, pp. 68–73.

[5] N. Chirdchoo, W. Soh, and K. C. Chua, “MACA-MN: A MACA-based MAC protocol for underwater acoustic networks with packet train for multiple neighbors,” in *Proc. IEEE VTC-Spring*, Singapore, May 2008.

[6] S. Azad, P. Casari, and M. Zorzi, “On ARQ strategies over random access protocols in underwater acoustic networks,” in *Proc. IEEE/OES Oceans*, Santader, Spain, May 2011.

[7] J. G. Proakis, E. M. Sozer, J. A. Rice, and M. Stojanovic, “Shallow water acoustic networks,” *IEEE Commun. Mag.*, vol. 39, no. 11, pp. 114–119, Nov. 2001.

[8] “Aquacomm: Underwater wireless modem.” [Online]. Available: http://www.dspcomm.com/products_aquacomm.html

[9] “Underwater acoustic modem models.” [Online]. Available: http://www.link-quest.com/html/models1.html

[10] “S2C R 48/78 underwater acoustic modem.” [Online]. Available: http://www.evologics.de/en/products/acoustics/s2cr_48_78.html

[11] P. Casari and M. Zorzi, “Protocol design issues in underwater acoustic networks,” *Elsevier Comput. Commun.*, vol. 34, pp. 2013–2025, June 2011.

[12] Y. Xiao, Ed., *Underwater Acoustic Sensor Networks*. Auerbach publications, 2008, ch. MAC Protocol Design for Underwater Networks: Challenges and New Directions, by V. Rodoplu and A. A. Gohari.

[13] M. Gao, W.-S. Soh, and M. Tao, “A transmission scheme for continuous ARQ protocols over underwater acoustic channels,” in *Proc. IEEE ICC*, Dresden, Germany, June 2009.

[14] L. Badia, P. Casari, M. Levorato, and M. Zorzi, “Analysis of an automatic repeat request scheme addressing long delay channels,” in *Proc. AINA*, Bradford, UK, May 2009.

[15] B. Tomasi, J. Preisig, and M. Zorzi, “On the predictability of underwater acoustic communications performance: the KAM11 data set as a case study,” in *Proc. ACM WUWNet*, Seattle, WA, 2011.

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

[17] R. Urick, *Principles of Underwater Sound*. NY: McGraw-Hill, 1983.

[18] F. Guerra, P. Casari, and M. Zorzi, “World Ocean Simulation System (WOSS): A simulation tool for underwater networks with realistic propagation modeling,” in *Proc. ACM WUWNet*, Berkeley, CA, Nov. 2009.

[19] B. Peleato and M. Stojanovic, “Distance aware collision avoidance protocol for ad hoc underwater acoustic sensor networks,” *IEEE Commun. Lett.*, vol. 11, no. 12, pp. 1025–1027, Dec. 2007.

[20] A. Syed, W. Ye, and J. Heidemann, “Comparison and evaluation of the T-Lohi MAC for underwater acoustic sensor networks,” *IEEE J. Sel. Areas Commun.*, vol. 26, pp. 1731–1743, Dec. 2008.

[21] N. Baldo, M. Miozzo, F. Guerra, M. Rossi, and M. Zorzi, “MIRACLE: The multi-interface cross-layer extension of ns2,” *EURASIP J. Wireless Commun. Netw.*, Jan. 2010. [Online]. Available: http://www.hindawi.com/journals/wcn/2010/761792/cta/

[22] M. Porter *et al.*, “Bellhop code.” [Online]. Available: http://oalib.hlsresearch.com/Rays/index.html

[23] “World ocean database.” [Online]. Available: http://www.nodc.noaa.gov/OC5/WOD09/pr_wod09.html

[24] B. Liang and Z. J. Haas, “Predictive distance-based mobility management for PCS networks,” in *Proc. IEEE INFOCOM*, New York, NY, Mar. 1999, pp. 1377–1384.

**Saiful Azad** received his PhD in Information Engineering from the University of Padova, Italy, in 2013. He completed his BSc in Computer and Information Technology at IUT, Bangladesh and his MSc in Computer and Information Engineering at IIUM, Malaysia. After the completion of his PhD, he joined the Department of Computer Science at the American International University–Bangladesh (AIUB) as a faculty member. His work on underwater acoustic networks started during his PhD program and is still his main research focus. His interests also include the design and implementation of communication protocols for different network architectures, QoS issues, network security, and simulation software design.

**Paolo Casari** [M’08] received the PhD in Information Engineering in 2008 at the University of Padova, Italy, where he is currently a postdoctoral research fellow. He has been actively researching cross-layer protocol design for MIMO ad hoc networks and wireless sensor networks. After spending a period at the Massachusetts Institute of Technology in 2007, he started working on underwater acoustic networks, which is currently his main focus. He has been the technical manager of the Italian projects WISE-WAI and NAUTILUS, and is currently involved in several efforts funded within the European Community’s FP7 program, and related to underwater acoustic networking. He served in the organizing committee of several conferences, and has been guest editor for the Hindawi Journal of Electronics and Computer Engineering special issue on “Underwater Communications and Networking.” His research interests include many aspects of underwater communications, such as channel modeling, network performance evaluation, cross-layer protocol design and at-sea experiments.

**Michele Zorzi** [F’07] received his Laurea and PhD degrees in electrical engineering from the University of Padova in 1990 and 1994, respectively. During academic year 1992-1993 he was on leave at UCSD, attending graduate courses and doing research on multiple access in mobile radio networks. In 1993 he joined the faculty of the Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy. After spending three years with the Center for Wireless Communications at UCSD, in 1998 he joined the School of Engineering of the University of Ferrara, Italy, where he became a professor in 2000. Since November 2003 he has been on the faculty of the Information Engineering Department at the University of Padova. His present research interests include performance evaluation in mobile communications systems, random access in mobile radio networks, ad hoc and sensor networks, energy constrained communications protocols, and underwater communications and networking.

He was Editor-In-Chief of IEEE Wireless Communications from 2003 to 2005 and Editor-In- Chief of the IEEE Transactions on Communications from 2008 to 2011, and serves on the Editorial Board of the Wiley Journal of Wireless Communications and Mobile Computing. He was also guest editor for special issues in IEEE Personal Communications (“Energy Management in Personal Communications Systems”) and IEEE Journal on Selected Areas in Communications (“Multimedia Network Radios” and “Underwater Wireless Communications and Networking”). He served as a Member-at-Large of the Board of Governors of the IEEE Communications Society from 2009 to 2011.