Wireless Networks (2021) 27:1941–1963
https://doi.org/10.1007/s11276-021-02550-0

A cooperative protocol for pervasive underwater acoustic networks

Lucas S. Cerqueira[1] · Alex B. Vieira[1] · Luiz F. M. Vieira[2] · Marcos A. M. Vieira[2] · José A. M. Nacif[3]

Accepted: 15 January 2021 / Published online: 5 February 2021

Abstract
The novel Underwater Wireless Sensor Network (UWSN) can contribute to monitor and explore aquatic environments. But, communicating in these environments is still hard and has many challenges. For example, optical and electromagnetic waves deteriorate from high-attenuation. Moreover, acoustic communication has a large packet error rate and low throughput. A large number of solutions to improve aquatic communication refers to routing protocols, medium access control protocols, and designing acoustic modems. Cooperative communication explores the broadcast nature of wireless transmission and enhances its performance. However, cooperative communication has not been fully explored in UWSNs. In this work, we present COPPER, a Cooperative Protocol for Pervasive Underwater Acoustic Networks. COPPER considers LLC and MAC sub-layers and operates synchronously or asynchronously over Time Division Multiple Access using a selective repeat ARQ scheme. COPPER exploits the broadcast nature of wireless communication and, sensor nodes that are idle can operate as a relay, enhancing communication by space diversity. Simulation results show that COPPER improves network performance. For example, the network goodput improves by 17% and the packet error rate decreases by 65%.

Keywords Sensor networks · Underwater · Cooperative communication · MAC

1 Introduction

Water covers 75% of the surface of our planet. Rivers, lakes, seas, canals, and oceans are paramount to life on Earth. Moreover, many social sectors require real-time monitoring of underwater resources. Underwater wireless sensor network (UWSN) is an area of research that can contribute to these sectors. In this sense, we can apply UWSNs in fields like offshore oil and gas extraction, oil spills, military surveillance and reconnaissance, mine detection, pollution monitoring, natural calamities like tsunami and hurricane forecast, coral reef and habitat monitoring of marine life, and fish farming [4, 27, 33].

For example, [15, 32, 57, 58, 71, 82] present systems [77, 78] to monitor the water quality, from fish farms to freshwater reservoirs. In systems like these, underwater sensors mostly observe water quality indicators, as temperature, dissolved oxygen, and pH, as well as pollution indicators as turbidity, ammonium, and nitrogen.

Large underwater sensor networks monitor oceans and coral reefs [1, 9]. For example, a UWSN monitors the Great Barrier Reef (GBR) off north-eastern Australia, with a focus on the factors that contribute to coral bleaching [9]. Underwater sensors are also widely used in real systems to monitor floods [61, 72] and oil spills [39, 47]. The petroleum industry utilizes UWSNs to help explore gas and oil in a wide range of applications, in special, when exploring gas and oil in the ocean.

✉ Alex B. Vieira
alex.borges@ufjf.edu.br

Lucas S. Cerqueira
lucas.saar@ice.ufjf.br

Luiz F. M. Vieira
lfvieira@dcc.ufmg.br

Marcos A. M. Vieira
mmvieira@dcc.ufmg.br

José A. M. Nacif
jnacif@ufv.br

[1] Department of Computer Science, Universidade Federal de Juiz de Fora, Juiz de Fora, MG, Brazil

[2] Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brazil

[3] Science and Technology Institute, Universidade Federal de Viçosa, Florestal, MG, Brazil

More recently, we note a growing interest in the development of underwater sensor nodes, which are self-sustainable. For example, Jang et al. [42] present an underwater sensor node that is capable of harvesting energy from underwater acoustic signals using piezoelectric interfaces and communicates by modulating the piezoelectric impedance.

In general, UWSNs present a collection of devices (sensor nodes). These devices contain several sensors, to collect several aquatic environment characteristics, such as water temperature, local pressure, water salinity, etc. Once a sensor node collects underwater environment information, it sends the data to a central node—in most cases using an ad-hoc network—which will pre-process and group data of all sensor nodes it receives, and then send to a far computational center—in most cases using a satellite link or high power radio—where data will be further processed.

Despite the importance of UWSN, it is still a difficult task to monitor these environments. The underwater environment enforces serious constraints for aquatic communication. Indeed, optical and electromagnetic waves deteriorate from attenuation and absorption under even small distance under water [75]. Besides, optical waves suffer high scattering, which turns its use not suitable for medium and long-distance communication. Therefore, radio-frequency (RF) and visible light technologies are not appropriate for deploying networks in an aquatic environment [50].

In this sense, acoustic communication has been widely adopted by academia and industry as a standard to underwater communication [38, 78]. But, this technology also presents its challenges. There are three major problems. First, bandwidth is limited and depends on distance. Second, it suffers from multipath fading that varies over time. Finally, sound speed is lower than electromagnetic waves [4, 56, 69]. Furthermore, UWSNs, like Wireless Sensor Networks (WSNs), are constrained by the energy which highlights the three above mentioned issues [3, 81].

Clearly, due to the acoustic channel particularities, the well-known wireless techniques to RF channels cannot be directly applied to UWSN [20]. As a consequence, to mitigate UWSN problems and provide better communication, some studies have been developed from the physical layer to the network layer. There are some work about developing acoustic modems to effectively utilize the channels [28, 62, 65], medium access control (see Sect. 2), and routing information between sensor nodes [10, 22–26, 54, 76, 84].

In such a scenario, the use of redundancy techniques, like Automatic Repeat Request (ARQ) protocols [67], and Forward Error Correction (FEC) is mandatory to achieve no error in communication. However, the use of FEC is not enough, since a channel with a high bit error rate (BER) will result in successive retransmissions [44]. Moreover, for protocols based on ARQ (e.g. Selective Repeat, Stop & Wait), given the high BER and long delay time of underwater channels, it is hard to achieve efficient throughput [34]. Furthermore, protocols based on ARQ have the disadvantage of higher energy consumption and higher latency.

Cooperative communication explores the broadcast nature of wireless transmission. While a node listens to other transmissions, it may act as a relay [51]. This communication mechanism might increase throughput, improve robustness, decrease packet loss, decrease network latency. Indeed, the relay may intensify the signal when transmitting in conjunction with the original node, or retransmit the original message if it has failed, allowing the original node to continue its transmission, leaving the task of retransmission to its partners. While the first approach mainly reduces the error rate, the second increases the flow rate. This form of spatial diversity is known as cooperative diversity or cooperative communication [45, 51].

In this work, a COoperative Protocol for PERvasive Underwater Acoustic Networks (COPPER) is presented. COPPER exploits that wireless communication is broadcasted by nature. Each sensor node can overhear other packets. It works as a relay by retransmitting to the sink node the packet it receives from others. By doing this, path diversity [51] is increased, which also improves the probability of successfully transmitting. Moreover, due to cooperation, time diversity can also be achieved because the overheard packet can be transmitted in a different period. When the source node transmits a new message, meanwhile the relays are recovering the last lost packet, which enhances the overall network throughput.

COPPER considers Logical Link Control (LLC) and Medium Access Control (MAC) data link sub-layers. It can operate synchronously or asynchronously over Time Division Multiple Access. It also utilizes selective repeat ARQ. COPPER takes into account sensor nodes that are idle to improve cooperative (space) diversity. COPPER differs from current approaches [6, 7, 34, 52, 53] since it integrates cooperative communication and selective repeat ARQ scheme for a collision-free MAC protocol.

Note that, when compared to existing opportunistic routing protocols (OR), we also explore the broadcast nature of wireless transmission to enhance communication performance. However, we do not focus on routing. Nodes cooperate to send data to a sink node directly. In other words, wireless sensor networks opportunistic routing is a new routing paradigm that chooses the closest node (according to a given metric) to the target node for forwarding the data [41]. In our protocol, we always forward data to a

given sink. Data may be forwarded concurrently in a cooperative way, exploring the idle nodes of the system.

Differently from our previous work [17, 18], where we have introduced the basic concepts of our cooperative communication protocol, in this work, we also explore the distance between underwater nodes and the sink node to properly schedule data transmission. The data transmission scheduling reduces the transmission collision and, as a consequence, enhances the underwater data communication. Moreover, when compared to our previous work, we properly formulate the problem by modeling the UWSN. Finally, we further evaluate all three versions of COPPER, presenting a deep discussion about system performance, system energy consumption, protocol overhead, and system scalability.

COPPER has been evaluated using the widely adopted ns-3 [59] simulator. Our simulation considers the case where many nodes send data to a sink. We vary network nodes' density and transmission rate. We compare COPPER to a baseline protocol without the cooperative mechanism. In general, COPPER improves the performance of the network. For example, the packet error rate is decreased by up to 65% in the best case, and goodput improves by 17% while consuming <1% more energy.

The remainder of this article is organized as follows: we present the state-of-the-art in Sect. 2. In Sect. 3, the problem formulation is provided. We describe the design and implementation of COPPER in Sect. 4. We detail the evaluation scenario and also show the results in Sect. 5. Finally, in Sect. 6, we present the conclusions.

2 Related work

Most of the existing works in the literature evaluate the effectiveness of cooperative transmission when applied to underwater wireless networks. Practically all of the existing works are an extension of cooperative protocols from existing (i.e., terrestrial) wireless networks [13, 14, 36, 37, 74, 79]. Just to illustrate, Carbonelli et al. [14] evaluate the energy efficiency of cooperative communication in a UWSN, using a multi-hop sensor network.

Some works propose a relay-aided protocol on the physical layer. For example, Vajapeyam *et al.* [74] proposed a protocol where relays work using an amplify-and-forward scheme. One may expect that amplify-and-forward, on the physical layer, also enhances the noise of the channel, which in turn, could make communication even worse. However, Han et al. [36] have shown that this scheme enhances the transmission quality, even under the presence of noise, and its amplification, on the physical layer. Differently, in this work, we propose a protocol that considers the Logical Link Control (LLC) and the Medium Access Control (MAC) data link sub-layers. It can operate synchronously or asynchronously over Time Division Multiple Access. It also utilizes selective repeat ARQ.

The outcomes of existing decode-and-forward, amplify-and-forward, and estimate-and-forward approaches have been previously evaluated by Han et al. [37]. They have also proposed an amplify-and-forward based protocol, namely Wave Cooperative. As expected, they have shown that their cooperative protocol achieves better performance than a system without any cooperative technique.

Wang et al. [79] presented a cooperation protocol that works asynchronously. Their protocol can be applied to a scenario presenting variable and large delay of message propagation. They have compared existing approaches to their technique, for example, decode-and-forward, direct transmission, and amplify-and-forward. They have shown that the improvement of each technique is highly related to the signal to noise ratio (SNR) of the channel. For instance, direct transmissions are better applied to scenarios with optimal SNR conditions. The amplify-and-forward scheme, on the other hand, presents a better performance when the SNR condition is not good.

There are several variations of the AF scheme. For example, Javaid [43] proposes RBCRP: a region-based cooperative routing protocol based on the amplify-and-forward technique over Rayleigh fading channels in underwater wireless sensor networks. Different from other approaches, they split the UWSN into smaller regions, clustering relays, and source nodes. They also use other techniques as energy harvesting and mobile sinks nodes, increasing the network lifetime, maximizing the overall throughput.

Doosti et al. [31] consider a perfect channel state information (CSI) at the source, relay, and destiny. They propose an amplify-and-forward (AF) relay protocol. In this sense, they model the UWSN and the relay selection as an optimization problem where the objective is to minimize the BER (bit error rate) and maximize the overall system capacity.

Ahmad et al. [2] also use, at the relay, an amplify-and-forward method, to perform cooperative communication. At the receiver, they also apply a fixed ratio combining strategy. Their proposal also explores the existence of physical layer diversity to perform cooperative communication and routing at the network layer. They then

formulate a routing problem to minimize energy usage on the UWSN. Yu et al. [83] also focus on energy. In this sense, they use mobile relay nodes to aid the source and sink nodes communication process. In their work, they properly select the mobile relay to cooperate taking into account the distance between nodes (i.e., the link distance) and residual energy each transmitting node has.

Li et al. [55] also address the relay selection. They model the relay selection using an extension of the multi-armed bandit problem which considers contextual information (i.e., contextual bandit problem). Instead of statistical or instantaneous information of the channel state, as error rate, they use well-known context information, as the position of nodes, including the relay.

A number of works use cooperative transmissions and ARQ schemes, jointly [34, 52, 53]. For example, Lee et al. [52] consider a single-hop underwater acoustic channel and propose a cooperative Stop-and-Wait ARQ scheme. According to their scheme, a final node that receives a packet with error requests this packet retransmission for a relay (the cooperative node). The closest node to her will be chosen as a relay. Authors, in this case, assume that system nodes can know their own location, and then, calculate the distance to their neighbors. Further, Lee et al. also have addressed cooperative communication in a multi-hop network. In this case, their protocol explores spatial diversity and enhances the communication process. Ghosh *et al.* [34] suggest a Hybrid ARQ in a cooperative protocol. They consider the use of FEC and redundant FEC bits to treat errors. Despite the importance of these three beforementioned works, we highlight that they only consider simple scenarios. For instance, only a single node starts the communication process. Moreover, the authors ignore or assume as resolved a number of MAC issues.

A cooperative ARQ scheme that works together with a MAC-based handshake protocol has been proposed by Kim et al. [46]. The handshaking mechanism they propose is based on a well-known mechanism, as RTS (Request to Send) and CTS (Clear to Send). The cooperative information of the protocol is transmitted through the handshaking mechanism. This process improves the underwater communication process. However, their protocol presents a high overhead due to the considerable number of messages of control. This high number of additional messages may increase the handshaking duration and increase the number of collisions in the system. As a consequence, they may experience lower system throughput depending on the system conditions. Finally, we are aware of few works which propose cooperation for systems based on OFDM and MIMO [63, 70]. These proposals work on the physical layer and, they disregard any ARQ scheme.

In sum, previous works are mostly related to the physical layer and explore amplify-and-forward mechanisms. Differently, our approach treats cooperative communication on the MAC layer. Note that, our approach can be used together with existing mechanisms and, in this sense, can turn the communication process even more effective. Our proposal also allows more than one communication node. In this sense, our proposal effectively coordinates the retransmission of messages exploring the capabilities of a TDMA based underwater sensor network. Moreover, our protocol can choose system relays dynamically and it can operate in an asynchronous or synchronous way. Finally, in contrast to our previous work [17, 18], we highlight that in this work, we explore the distance between underwater nodes and the sink node to properly schedule data transmission. The message scheduling avoids communication collision at the sink node, which in turn enhances the system throughput and reduces the overall error rate.

3 Wireless underwater network formulation

In this work, we consider a wireless underwater network with n nodes, including the sink node. Sensor nodes contain several sensors, to collect several aquatic environment characteristics, such as water temperature, local pressure, water salinity, etc. Once a sensor node collects underwater environment information, it sends the data to the sink node. The sink, in turn, does not generate data. It will pre-process and group data of all sensor nodes information, and then send them to a far computational center—in most cases using a satellite link or high power radio–, where data will be further processed. Moreover, the sink node acknowledges—or not– data packets it receives, transmitting ACK/NACK messages, respectively.

Figure 1 shows an example of UWSN. The nodes are scattered around the area of interest. These sensor nodes may be fixed to the bottom of the aquatic environment or

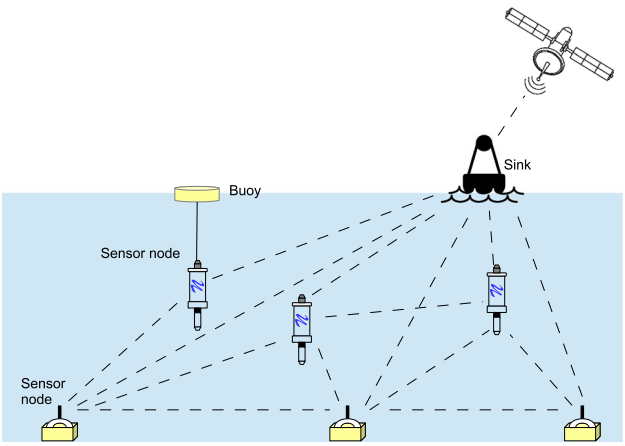


Fig. 1 Example of UWSN with sensor nodes and wireless links

supported by buoys. In some cases, underwater vehicles may transport information from one sensor node to another. Sink node, in turn, usually presents higher processing and energy capacity. This special node is capable of performing long-distance communication. In other words, the sink node gathers sensor nodes data and sends it to an external network where data will be further processed.

Several real systems present a scenario very close to the one we consider in this work. For example, systems that monitor oceans and coral reefs [1, 9] usually deploy underwater sensor nodes that capture water properties like temperature and salinity. These sensor nodes transmit data to a central node, usually deployed in a buoy. This central node, with larger computational capacity, transmits data to the internet (thought satellite link, for example). A large UWSN—with these characteristics— monitors the Great Barrier Reef (GBR) in northeastern Australia, focusing on the factors that contribute to coral bleaching [9]. Finally, we highlight the existence of recent projects, as the NEMO project [16], which uses acoustic communication and underwater networking technologies to environmental monitoring and the automation of underwater fish farms.

We opt for a time division multiple access (TDMA) data link. In this case, every node receives a time slot with exclusive access to the communication channel. Figure 2 illustrates time period assignments for a system formed by n sensor nodes. The interval composed of n time periods, one of each node, is called a frame, being repeated indefinitely.

In the physical layer, we consider modems that operate on *half-duplex*, that is, they can not receive and transmit data packets at the same time. Thus, while one node is transmitting, any received data is discarded. Full-duplex modems are less common in underwater networks because of the cost (multiple antennas), complexity, and channel capacity reduction [35].

We consider the nodes immovable and distributed randomly. They form a star topology within an area of 200 m x 200 m, and 70 m deep. The collecting node, however, is always positioned in the middle of the cube. Therefore, all messages are addressed towards the sink node. This topology is common in RSA once it simplifies MAC protocol configuration and provides good results for small networks [44].

To simulate the aquatic acoustic channel we use widely adopted error models. These models are valid for a depth of approximately 100 m, called shallow water. First, for every two nodes far apart d meters, we compute the acoustic signal attenuation with the following equation [73]:

$$10\log A(d,f) = k \cdot 10\log d + d \cdot 10\log\, a(f), \tag{1}$$

where f is the transmission frequency. The initial term represents the dispersion of the signal loss, with the coefficient k being known as dispersion factor, which indicates how the wave spreads through the medium. For values of $k = 1.0$ the wave spreads like a cylinder where the center is the point of origin of the signal, and similarly, for $k = 2.0$ the wave spreads like a sphere. The value of $k = 1.5$ represents a middle ground between the two dispersions and is the value commonly used [60, 68]. The second term is absorption loss, which represents how much energy is lost in the form of heat as it propagates through the medium. The absorption coefficient ($a(f)$) is calculated by the Thorp approximation [11]:

$$\begin{aligned} 10\log\, a(f) =& 0.11\,\frac{f^2}{1+f^2} + 44\,\frac{f^2}{4100+f} \\ &+ 2.75\cdot 10^{-4} f^2 + 0.003, \end{aligned} \tag{2}$$

which is valid for frequencies $f \geq 0,4$ Hz.

Figure 3 shows how dispersion and absorption affect signal attenuation. We calculate the dispersion loss using $k = 1.5$, while the absorption loss is calculated for frequencies $f = 4$, 10, 20, and 60 kHz. For lower distances, the dispersion has a greater impact on attenuation, being dominant even at higher frequencies [12]. In the case of the scenario we consider, where the maximum distance

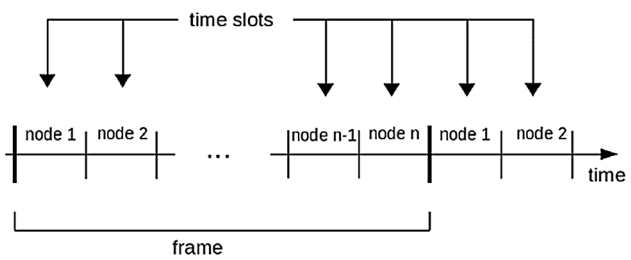


Fig. 2 TDMA scheme frames and time intervals

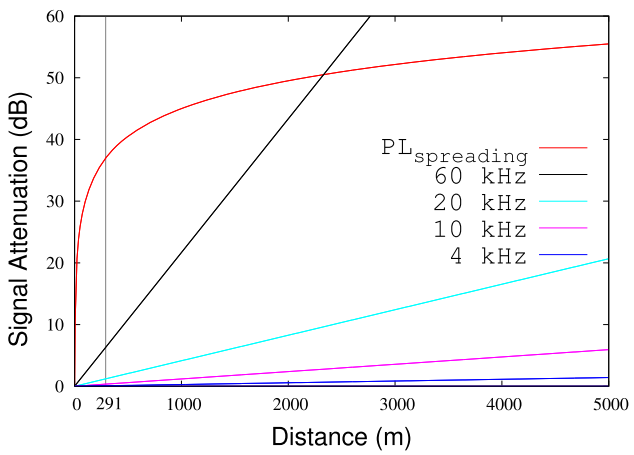


Fig. 3 Attenuation signal components: dispersion and absorption

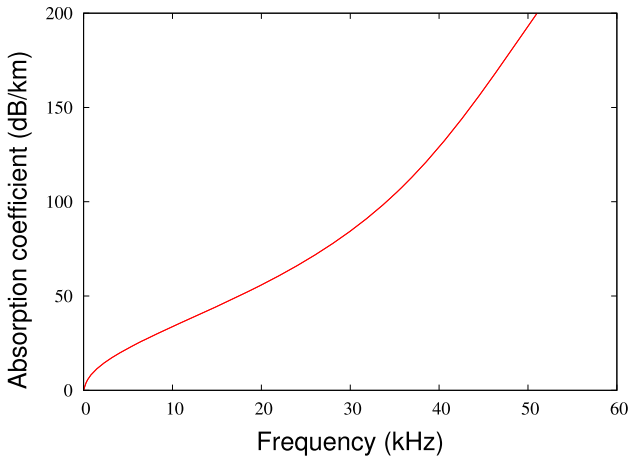


Fig. 4 Absorption coefficient with respect to frequency

between the sink node and a sensor node is no longer than 158 m, the dispersion presents an impact on signal attenuation in about 30 dB. Note that the absorption impact in this same scenario, even for 60 KHz, is lower than 5 dB.

Figure 4 shows the absorption coefficient (Eq. 2) with respect to frequency. The absorption coefficient rapidly increases with the frequency, which clearly indicates that the use of higher frequencies in UWSNs communication may not be feasible. Analogously, the conjunction effect of frequency and distance also restricts underwater communication to short distances and low frequencies, e.g. frequencies ranging from 1 to 50 kHz and up to 200 m sensor distances, which is close to the scenario we evaluate in this work.

Thus, it is possible to compute the SINR as:

$$\gamma(d) = SL - TL - NL + DI, \tag{3}$$

where SL is the transmitted signal energy, TL is the signal loss along the transmission, calculated in Eq. 1, DI is the directive factor, which for acoustic modems is given for $DI = 0$ NL represents the noise level of the ambient, which is an approximation calculated by the Wenz equation [80] in terms of a frequency f:

$$NL(f) = N_t(f) + N_s(f) + N_w(f) + N_{th}(f). \tag{4}$$

The equation is composed of the main noises present in the maritime environment. The term N_t is the noise of turbulence generated by the agitation of water medium. N_s means the noise caused by vessels, N_{th} is the thermal noise, and N_w is wind noise. The power spectral density function of each component is modeled by:

$$\begin{aligned} 10\log N_t(f) &= 17 - 30\log f \\ 10\log N_s(f) &= 40 + 20(s - 0.5) + 26\log f \\ &\quad - 60\log(f + 0.03) \\ 10\log N_w(f) &= 50 + 7.5w^{1/2} + 20\log f \\ &\quad - 40\log(f + 0.4) \\ 10\log N_{th}(f) &= -15 + 20\log f. \end{aligned} \tag{5}$$

In the $N_s(f)$ equation, the term s is called a shipping factor and represents naval activity. The value of s varies from 0, indicating no naval activity, up to 1, indicating intense activity. The noise coming from boats is predominant at low frequencies: from 10 to 100Hz, and is little influential in the frequencies commonly used in RSA. Similarly, the thermal noise $N_{th}(f)$ is meaningful only at very high frequencies, over 100 kHz. In the noise $N_w(f)$, w represents the speed of wind measured in m/s. Wind is the main source of noise in the frequency range of 100 Hz to 100 kHz, which is the frequency band utilized by most modern acoustic modems.

Lastly, it is possible to express the probability of a bit error using the modulation BPSK [64], as:

$$p_e(d) = \frac{1}{2}\left(1 - \sqrt{\frac{\Gamma(d)}{1 + \Gamma(d)}}\right), \tag{6}$$

where $\Gamma(d)$ is given for:

$$\Gamma(d) = 10^{\gamma(d)/10}. \tag{7}$$

Thus, we consider that in a network of aquatic sensors where the modems transmit at a certain frequency f, the probability of bit error varies according to the length d between the sending and receiving nodes.

4 Cooperative communication proposal

As we stated in Sect. 3, in this work, we consider a UWSN where $n - 1$ sensor nodes transmit their data to a sink node. Moreover, the sensor network employs a TDMA medium access protocol. Also, recall that UWSN transmission has a broadcast-like nature. In this case, a sensor node sends a packet to the sink node and all other nodes in its neighborhood may receive the data packet it transmits.

For instance, suppose a given source node O trying to transmit to the sink node, namely S. Let R be any of the idle node positioned between O and S. When O attempts to transmit a data packet to S, node R is more suited to obtain the correct packet than S. In fact, because of spatial correlation and channel fading, the error probability between $O \rightarrow S$ is larger than the error probability between $O \rightarrow R$.

Table 1 Main COPPER data structures

Data structure	Description
O	The source node O
S	The sink node
R	The idle node, which will cooperate
m_y	Data message y
Frame i	The current TDMA frame
Frame $i+1$	The next TDMA frame
DP	The data period of the TDMA frame
SP	The signaling period of the TDMA frame
Cooperation buffer	A buffer where idle nodes store data for further cooperation
BULK NACK	A map of all "NACKed" messages within the current frame by the sink node

In case S does not receive the data correctly, it may signal to O this error with a not acknowledgment (NACK) message. In this case, if R did receive data correctly, and R stored data in a temporary buffer, it may cooperate to O by offering R's slot of time to transmit the above-mentioned packet to S. As the link between $R \rightarrow S$ presents lower error probability then the original link $O \rightarrow S$, retransmission may occur with a higher success probability. Moreover, the source node S may use its consecutive timeslot to continue its communication process. As a consequence, cooperative communication may also increase overall system goodput.

In the following, we present COPPER. First, we defined a synchronized version of the protocol, where sensor nodes combine themselves to perform the cooperation (Sect. 4.1). Then, we present a more flexible and unsynchronized version of the protocol (Sect. 4.2). Finally, we combine COPPER into a data transmission scheduling mechanism (Sect. 4.3).

Table 1 summarizes the main data structures COPPER uses in this work. First, w.l.o.g., we assume 3 communicating devices, the source, the sink, and the relay node. Sensor nodes present buffers, where they store the messages they want to transmit. Messages are indexed, as data message m_1, m_2, etc. As we discussed in Sect. 3, we assume a TDMA like system and, within a Frame i, there is a signaling period SP and a data period DP. Sensor nodes present enough storage capacity to temporally save messages, in a *"Cooperation buffer"*. The *"Cooperation buffer"* has all messages a node can cooperate in the next TDMA frame. Finally, COPPER uses a BULK NACK, a special message which sink nodes broadcast during its data period. In this message, the sink nodes indicate all messages it incorrectly received during the current frame.

4.1 Synchronous COPPER (COPPER-S)

First, we designed a Synchronous COPPER. In each TDMA frame, there is a signaling period (SP), followed by a data period (DP). Sensor nodes utilize the SP period to communicate control messages, including the cooperation control messages. In other words, when a node perceives a cooperation opportunity, it signals its intention to retransmit the data packet in place of the transmitter node. All nodes will note which nodes are capable of cooperating and, as a consequence, they may easily elect a relay. Algorithm 1 details the Synchronous COPPER and present the signaling period (SP), followed by a data period (DP).

Algorithm 1: Synchronous COPPER()

```
int Frame_i = 0;   % TDMA frame counter
define MAXNODES;   % defines the number of underwater nodes, including sink
int TDMA_slots = MAXNODES;
define node <int CooperationBuffer[MAXNODES]: buffer dataBuffer>;
node nodes[MAXNODES];
int BULKNACK[MAXNODES];
while (true) do
    % TDMA SP - Signaling Period
    for (i = 1 to (TDMA_slots - 1) ) do
        % Case node i is idle, it can cooperate
        if (nodes[i].transmissionBuffer is empty) then
            % To cooperate, it sends WTC message with a buffer map
            nodes[i].signalize(node[i].CooperationBuffer[]);
        end
    end
    % TDMA DP - Data Period
    for (i = 1 to (TDMA_slots - 1) ) do
        if (nodes[i].transmissionBuffer has failed packet to retransmit &&
            #attempts < threshold) then
            #attempts++;[1]
            nodes[MAXNODES].SENDV(transmissionBuffer[]);
        else
            if (nodes[i].transmissionBuffer has dataPacket to transmit) then
                nodes[i].SENDV(transmissionBuffer[]);
            else
                if (nodes[i].cooperationBuffer has dataPacket to cooperate) then
                    if (nodes[i] is the eligible[2] relay for a given Packet p then
                        nodes[i].SENDV(cooperationBuffer[p]);
                    end
                end
            end
        end
        % During a slot from i, all other nodes can receive the packet
        for (j = 1 to (TDMA_slots - 1) | j != i) do
            dataPacket = nodes[i].recv();
            % Check dataPacket is Ok AND is not a retransmission
            if CHECK(dataPacket) && dataPacket.timeStamp == Frame_i then
                nodes[j].CooperationBuffer[j]=dataPacket;
            end
        end
        % Sink node receiving...
        dataPacket = nodes[MAXNODES].recv();
        % Sink builds BULKNACK only for current frame...
        if CHECK(dataPacket) && dataPacket.timeStamp == Frame_i then
            BULKNACK[j]=true;
        else
            BULKNACK[j]=false;
        end
        % Sink node consumes all correct packets...
        if CHECK(dataPacket) then
            nodes[MAXNODES].addBuffer(dataPacket);
        end
    end
    % The last DP slot belongs to the sink
    nodes[MAXNODES].SENDV(BULKNACK[]);
    % At this point, all nodes will receive the bulknack and remove from its
      cooperation buffer the data packets the sink received correctly.
    Frame_i++;
    % END of TDMA DP - Data Period
end
```

[1] Some control variables have been omitted for simplicity.

[2] In this work, the node closer to the sink (lower ID) will be elected as relay.

As shown in Algorithm 1, a node sends data during its time slot –during the DP– one data packet in a single frame. After that, if a node receives a correct packet that is addressed to the sink, this node saves this content in a particular buffer that stores only cooperation messages. When the data period ends, the sink node responds with an ACK or a NACK. NACK indicates that the current frame has not been received correctly. Subsequently, by receiving

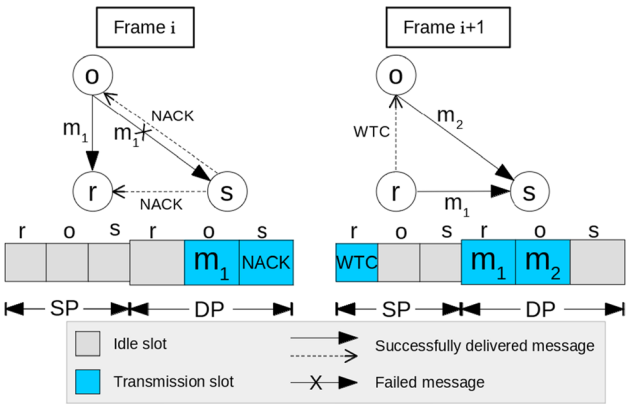


Fig. 5 A scenario with three nodes and the Synchronous COPPER protocol

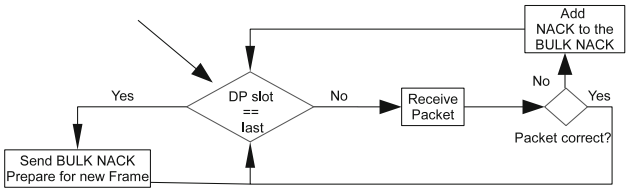


Fig. 6 COPPER-S: Sink node behavior

the NACK, each sensor node knows the failed packets. Then, the sensor nodes can cooperate in the next period by resending the missing message.

At the signaling period, nodes that are idle and can cooperate send a *Want To Cooperate* (WTC) message (24 bits message). This WTC message is used to indicate which packets a node has in its buffer and may retransmit in place of the original source. In case a collision occurs—i.e., more than one sensor node tries to cooperate–, sensor nodes may use a given policy to decide which of them will act as a relay. In this work, we assume the first node to send a WTC message will be select as a relay. We further discuss the policy we adopt in this work in the end of this section.

Thus, at the end of the SP, every node is aware of the message it will send in the current period and so, all nodes manage their own buffer of cooperation. Note that, this mechanism also allows the source to continue its transmission. Finally, as we are considering a cooperation period of one period, nodes allocate a cooperative buffer of size $n-1$ packet, where n is the size of the network.

Figure 5 depicts one scenario with 3 nodes, a source O, a sink S, and a relay node R. During a given frame i, the original source node O sends one message m_1, which is correctly received by relay R. This message is not properly received by the sink S. At the end of the frame, S sends a NACK message to signal the failure of the message transmission. This is received by all neighboring nodes in the network (in this case, O and R). If R is idle, it will send a WTC message to signal that it will cooperate in the next period. Thus, source O will send message m_2 during frame $i+1$, while R retransmits m_1.

Figures 6, 7 and 8 present flowcharts detailing the Synchronous COPPER behavior of the sink node, the sensor node—as a receiver—and the sensor node—as a transmitter—respectively. These flowcharts detail the distributed algorithm of our TDMA-based protocol.

According to Fig. 6, the **sink node**, when receiving a packet from a sensor node O, checks if the packet is correct. There are several mechanisms to check packet integrity on MAC and, our protocol is flexible enough to use any of them. If the packet is not correct, the sink node will add the packet's identifier to the NACK message to be broadcast at the sink's next data period. Note that we reserve the last slot of the TDMA frame for the sink node. In this sense, by the end of the current TDMA frame, the sink node will be the last to transmit. It then sends an aggregate message, which we call BULK NACK (40 bits message), informing the packets it has not correctly received. Then, the sink node clears its buffers and prepares for the next TDMA data frame.

Figure 7 details the Synchronous COPPER running in a receiver node. When receiving a data packet, the node checks if it is idle (i.e., it does not have data in its transmitting buffer) and also checks if it has correctly received the data. Moreover, the sensor node checks if the data does not have an identification belonging to the previous DP, which indicates this data is being retransmitted. Then, if yes (for all three questions), the node stores this data in the cooperation buffer for further processing. This node waits for the BULK NACK, which sink node sends at the end of DP.

Figure 8 details the Synchronous COPPER behavior of the **sensor node, as a transmitter**. As shown in this figure, during its DP slot, a sensor node may retransmit its own data, which has not been properly received by the sink node previously. In this case, the number of retransmission attempts is limited by a given threshold. If the sink node has properly received the packet, the sensor node may send its new packet (if it has data to transmit). Finally, if the sensor node is idle, it may act as a relay for the packet(s) it has stored in its cooperation buffer. Recall that all nodes send WTC messages during the SP. Then, a node checks all of these messages and, according to a given policy, it may locally decide if it is the eligible relay or not. In the current version of the protocol, the node with the lowest ID will act as the relay.

Figure 9 exemplifies another scenario in which four nodes communicate to one sink. At the time of the initial data period, the four nodes try to send their data packets to the sink. As shown in this figure, packets from nodes 3 and 4 failed, while packets from nodes 2 and 1 were well

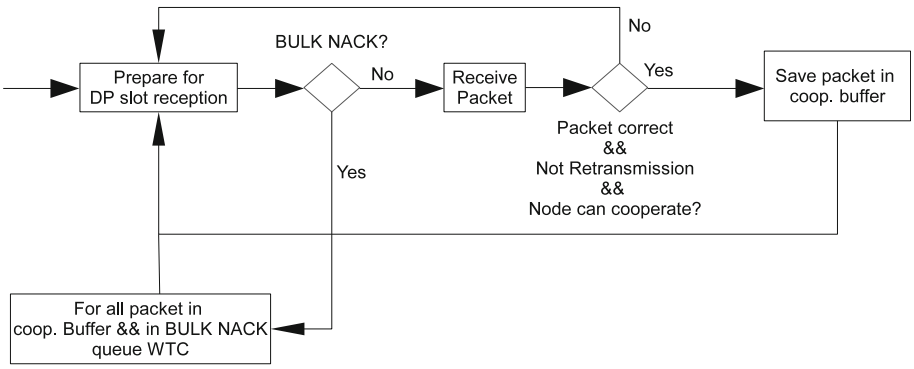


Fig. 7 COPPER-S: Sensor node behavior as a receiver

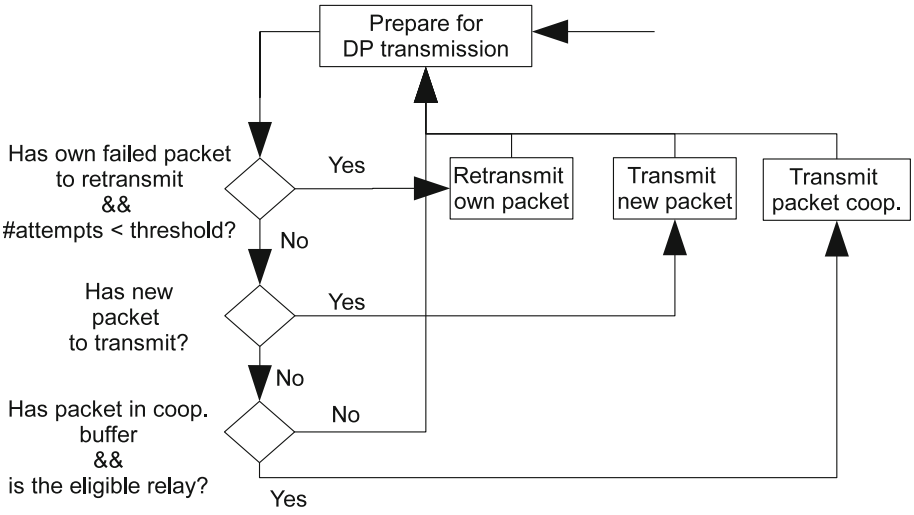


Fig. 8 COPPER-S: Sensor node behavior as a transmitter

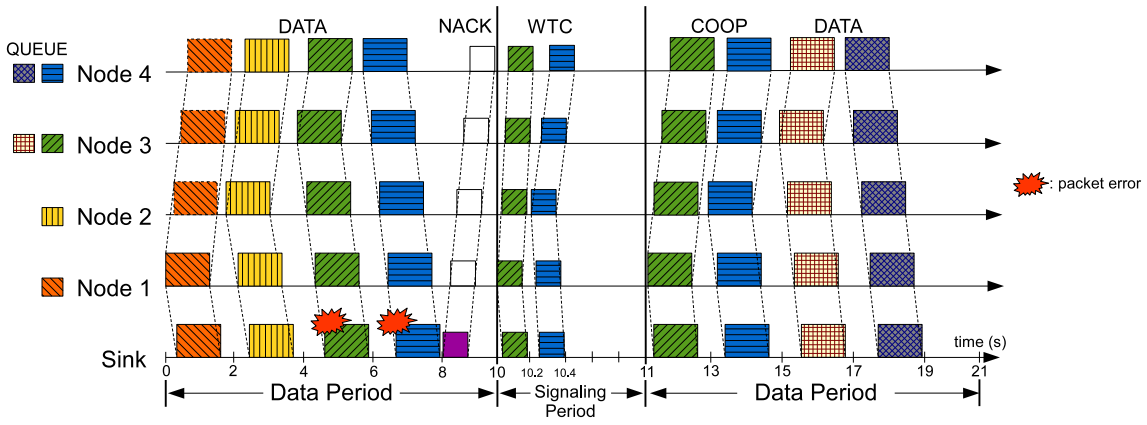


Fig. 9 A scenario with 5 nodes and the Synchronous COPPER protocol

received. Then, the sink node emits a NACK message, indicating that it did not accurately receive packets from nodes 4 and 3. When the nodes receive the NACK, they verify if there are any messages stored on their respective local cooperative buffers. If they have, they indicate their willingness to collaborate at the nearest signaling period. Note that nodes receive messages at different moments. For example, when node 1 sends its message, the sink and the sensor node 2 receive it almost at the same moment. Then, node 3 receives it and, finally, node 4. This difference between the time periods nodes receives data may be further explored to properly schedule data transmission order, avoiding collisions on the communication medium.

The synchronous COPPER assumes that nodes will combine themselves to avoid the duplication of data. In other words, distinct nodes will not cooperate

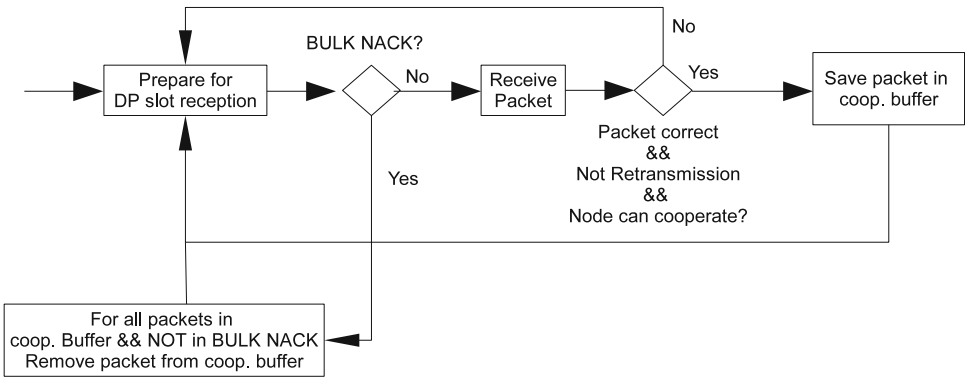


Fig. 10 COPPER-A: Sensor node behavior as a receiver

retransmitting the same message. In this work, we consider an ordering to eliminate the cooperation duplication. For instance, according to Fig. 9, the initial WTC message is from node 1. At the moment node 2 receives the node 1 WTC message, it realizes that node 1 will help by sending the node 3 message not received by the sink. Then, node 2 transmits a WTC message indicating its plan to cooperate in transmitting node 4 message. After receiving the WTC messages, nodes 3 and 4 are aware that they should send their following processed messages. In the following data period, the sink correctly receives the nodes 1 and 2 cooperative messages. Likewise, during this frame, nodes 3 and 4 successfully transmit data to the sink.

Choosing the relay node is vital for cooperation triumph. Naturally, the nearest the relay is to the sink; the higher are the odds of successful cooperation. Undoubtedly, the nearest nodes to the sink have a smaller PER than the nodes further away as the PER grows with the distance. According to our previous example, for a given failed message, the initial node to send a WTC message is selected as a relay. Thus, to order nodes by their PER, we arrange the sensor nodes time slots considering the distance to the sink, as done in [30]. That is, the initial time slot of each frame belongs to the nearest node to the sink. According to the COPPER-S protocol, when a node is closer to the sink, it has higher chances to become a relay.

A major drawback of this approach is the non-uniform energy utilization. Nodes that are near the sink may end up sending more packets than other sensor nodes, leading to unequal energy spending. We point out that we intend to study different relay selection policies as future work, including policies that consider node energy level/capacity.

4.2 Asynchronous COPPER (COPPER-A)

We have also designed and implemented an asynchronous version of COPPER, called *COPPER-A*, as shown in Algorithm 2. In COPPER-A, sensor nodes are not required to signal for cooperation. As a consequence, this version of COPPER is simpler than synchronous COPPER. Moreover, it has faster cooperation retransmissions since it does not require a signal period. But, sensor nodes do not know what frames each other sensor node would cooperate. This can lead to cooperation retransmissions being replicated. Indeed, Asynchronous COPPER presumes that when a sensor node overhears a NACK frame, it will randomly pick, from its cooperative buffer, one of the messages that had failed before. Clearly, there can be a replication of failed messages by many sensor nodes, which increases energy consumption. But, it also increases path diversity, which in turn, reduces PER.

Algorithm 2: Asynchronous COPPER()

```
1  int Frame_i = 0;   % TDMA frame counter
2  define MAXNODES;   % defines the number of underwater nodes, including sink
3  int TDMA_slots = MAXNODES;
4  define node <int CooperationBuffer[MAXNODES]: buffer dataBuffer>;
5  node nodes[MAXNODES];
6  int BULKNACK[MAXNODES];
7  while (true) do
8      % TDMA SP - Signaling Period
9      for (i = 1 to (TDMA_slots - 1) ) do
10         % Nodes do not need to send WTC message
11     end
12     % TDMA DP - Data Period
13     for (i = 1 to (TDMA_slots - 1) ) do
14         if (nodes[i].transmissionBuffer has failed packet to retransmit &&
15             #attempts < threshold) then
16             #attempts++;[1]
17             nodes[MAXNODES].SENDV(transmissionBuffer[]);
18         else
19             if (nodes[i].transmissionBuffer has dataPacket to transmit) then
20                 nodes[i].SENDV(transmissionBuffer[]);
21             else
22                 if (nodes[i].cooperationBuffer has dataPacket to cooperate) then
23                     nodes[i].SENDV(cooperationBuffer[RANDOM]);
24                 end
25             end
26         end
27         % During a slot from i, all other nodes can receive the packet
28         for (j = 1 to (TDMA_slots - 1) | j != i) do
29             dataPacket = nodes[i].recv();
30             % Check dataPacket is Ok AND is not a retransmission
31             if CHECK(dataPacket) && dataPacket.timeStamp == Frame_i then
32                 nodes[j].CooperationBuffer[j]=dataPacket;
33             end
34         end
35         % Sink node receiving...
36         dataPacket = nodes[MAXNODES].recv();
37         % Sink builds BULKNACK only for current frame...
38         if CHECK(dataPacket) && dataPacket.timeStamp == Frame_i then
39             BULKNACK[j]=true;
40         else
41             BULKNACK[j]=false;
42         end
43         % Sink node consumes all correct packets...
44         if CHECK(dataPacket) then
45             nodes[MAXNODES].addBuffer(dataPacket);
46         end
47     end
48     nodes[MAXNODES].SENDV(BULKNACK[]);
49     % At this point, all nodes will receive the bulknack and remove from its
         cooperation buffer the data packets the sink received correctly.
50     Frame_i++;
51     % END of TDMA DP - Data Period
52 end
```

Figures 10 and 11 present flowcharts detailing the protocol behavior for the sensor node—as a receiver—and the sensor node—as a transmitter—respectively. According to Fig. 10, a sensor node, when receiving a packet, checks if it is idle (i.e., it does not have data in its transmitting buffer) and also checks if it correctly received the data. By the end of the DP, the sink node sends a BULK NACK, informing the packets it has not correctly received. In the case of COPPER-A, the sensor node does not need to advertise its cooperation intention to cooperate. It only deletes the packets from its cooperation buffer which are not annotated in the BULK NACK message. Moreover, as shown in Fig. 11, the major difference from COPPER-A to COPPER-R, considering a sensor node as a transmitter, is that the sensor node does not check if it is eligible to transmit the packet cooperatively. It only transmits it, if it has in its cooperative buffer.

Figure 12 depicts an example of COPPER-A where four nodes are sending packets to the sink. This figure presents two consecutive data period. In comparison with Fig. 9,

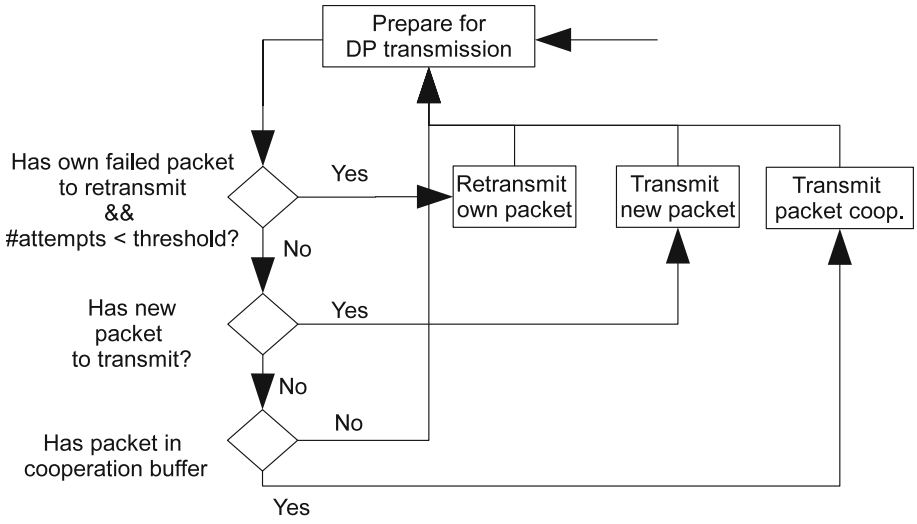


Fig. 11 COPPER-A: Sensor node behavior as a transmitter

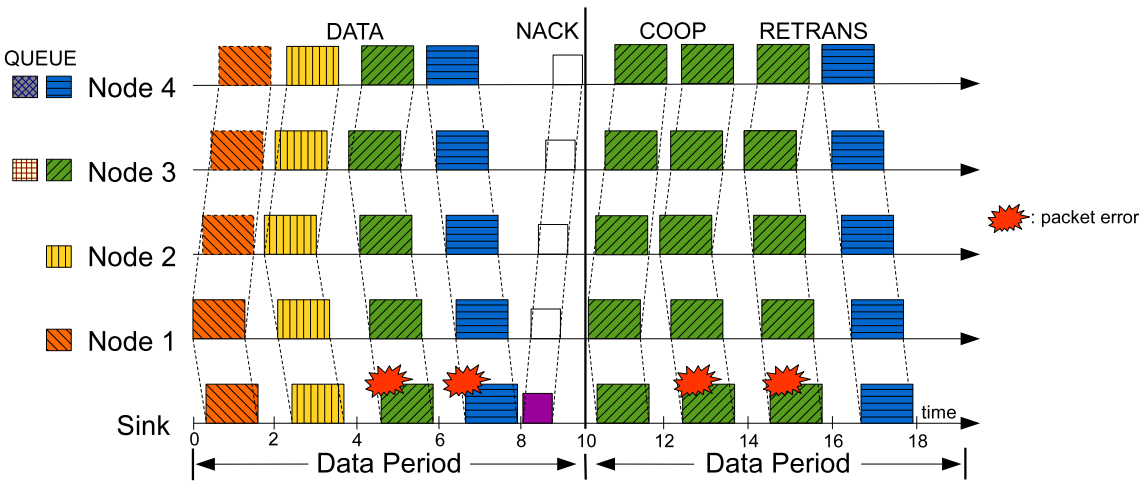


Fig. 12 Two frames of asynchronous COPPER

observe that there is no period for signaling anymore. In this new scenario, during the first frame, all nodes send messages to the sink. Again, in this example, the frames from sensor nodes 3 and 4 are not properly received in the sink node. Later, as expected, the sink node sends a NACK message indicating that the frames from sensor nodes 3 and 4 were not properly received.

During the second data period, sensor nodes 2 and 1 are idle, which means they are available to cooperate and act as relays. Since both —sensor nodes 1 and 2—got the NACK message and they have the missing messages on their cooperative buffer from overhearing, these nodes pick randomly one of the messages for cooperation. In this illustration, they select the same frame. Moreover, sensor node 3 also retransmits its failed frame, having 3 possible distinct ways ($1 \rightarrow s$; $2 \rightarrow s$; $3 \rightarrow s$). In this same data period, sensor node 4 transmits its failed message lonely.

Algorithm 2 further details the Asynchronous COPPER. Note that, in this case, all nodes send the data they want to cooperate with. In this work, in case they have multiple data in their cooperation buffer, we choose a data packet randomly.

4.3 Scheduled COPPER

Time division-based protocols, as TDMA, present a guard time between consecutive transmissions to avoid collisions at the sink node. The total guard time varies according to the distance from the sensor nodes to the sink. Indeed, the guard time is related to the largest propagation time between the sink node and the sensor node. The largest propagation time, D_{max}, can be obtained as shown in Eq. 8:

$$D_{max}(d) = \frac{d}{v}, \tag{8}$$

where v represents the speed of sound in the aquatic environment and d indicates the largest distance between the sink and a sensor node. In a square UWSN, the largest distance between any 2 sensor nodes corresponds to the diagonal size.

Despite the collision avoidance, this period of inactivity represents a waste of bandwidth. In fact, depending on the UWSN area, the time guard will impose a long period of inactivity on the sensor nodes' communication. For example, for a UWSN in which nodes are randomly disposed of in a 1000 m side square area, the largest distance between two distinct nodes is ≈ 1414.21 m. According to Eq. 8, the time guard after each frame is ≈ 0.94 s.

To overcome longtime guard periods, we propose the Scheduled COPPER, an extension of the Synchronous COPPER that incorporates the scheduling of the transmissions. According to the Scheduled COPPER protocol, each sensor node calculates its specific propagation delay, based on its distance to the sink node. Sensor nodes, then, can anticipate its transmission in a manner sensor node will receive the data in a proper period, without any collision with other sensor nodes messages. This approach aims at achieving 100% channel utilization by reducing receiver-side collision at the sink node, and has been successfully used by several works [5, 8, 19, 29, 40, 48, 49].

In this sense, Algorithm 3 presents the Scheduled COPPER Preparation Phase. In this phase, each node calculates the amount of time it has to anticipate DP transmission of the TDMA. Again, we assume, w.l.o.g., nodes know a priori the distance between then and the sink node and, nodes are assigned to TDMA slots according to this distance, in a crescent way. Once each node calculates the amount of time it has to anticipate its transmissions, they act according to the desirable cooperation policy. In this work, we assume they act similarly to the COPPER-S.

For example, if a sensor node is distant 1500 m from a sink node, the propagation delay between them will be 1 s. Thus, the sensor node must anticipate its transmission by 1 s so the data packet arrives at the sink node at the very beginning of the data period belonging to the original source node.

Figure 13 presents the traditional scenario, where sensor nodes do not schedule their transmission and insert a large time guard period. In contrast, Fig. 14 presents a scenario where sensor nodes calculate their propagation delay to the sink and then, anticipate their transmission. Clearly, when properly calculating the propagation delay and anticipating data transmission leads to the reduction of time guards and, in some cases, its complete removal.

Figure 15 presents a similar scenario to Figs. 9 and 12. In this figure, four nodes send data packets to a sink node. Note that the cooperation scheme operates exactly as it does in the Synchronous COPPER protocol. Nodes 3 and 4 fail to deliver their first messages due to channel error. Nodes 1 and 2 are free to cooperate in the next data period and retransmit packets from nodes 3 and 4. By anticipating the transmission of a message, collisions can occur at the sensor nodes. In other words, data transmission will be correctly aligned on the sink node but, a given sensor node may overhear two messages at the same time. As expected, some sensor nodes may not correctly receive other nodes' messages. As a consequence, the number of available nodes to cooperate for a given message may be reduced. Again, we emphasize that this type of collision does not affect the correct transmission and reception of messages at the sink node. It only reduces the number of possible cooperators.

5 Evaluating COPPER

In this section, we evaluate COPPER and show its performance. First, we present the evaluation methodology (Sect. 5.1) Then, we define metrics we use to compare the

Algorithm 3: Scheduled COPPER Preparation Phase()

```
1 define MAXNODES;
2 define v; % speed of sound in water 1500
3 define node <int CooperationBuffer[MAXNODES]: buffer dataBuffer:
4                 float d: float antecipateTDMAPeriod >;
5 node nodes[MAXNODES];
6 for (i = 1 to (MAXNODES - 1) ) do
7 |   nodes[i].antecipateTDMAPeriod = nodes[i].d / v;
8 end
```

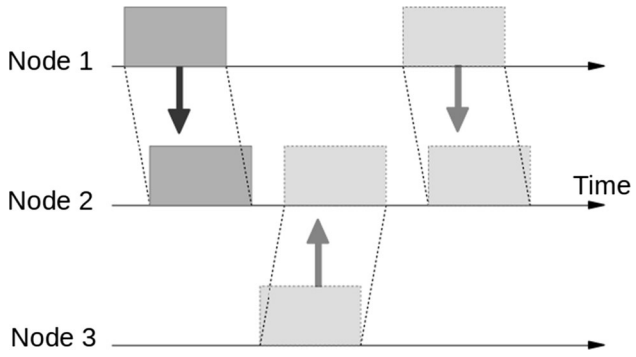


Fig. 13 Traditional scenario, with larger time guard

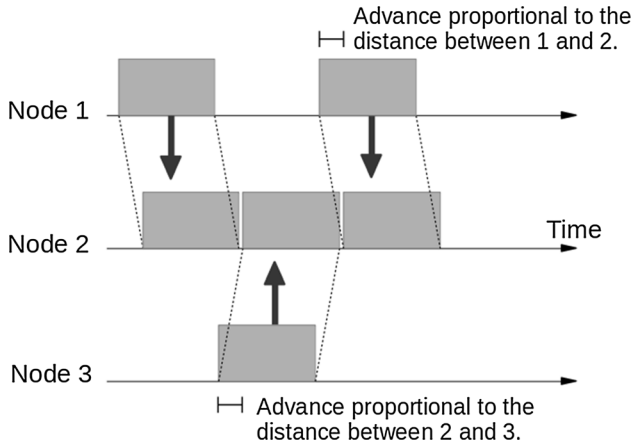


Fig. 14 Scheduled COPPER transmission diagram

protocols we consider in this work (Sect. 5.2). Finally, we discuss the evaluation results (Sect. 5.3).

5.1 Evaluation methodology

COPPER is evaluated with ns-3 (network simulator) version 3.27 [59], which is a discrete event simulator. We implemented the asynchronous COPPER, the synchronous COPPER, the scheduled COPPER, and a Time Division Multiple Access as a baseline protocol without any cooperation mechanism.

Ns-3 provides an algorithm on flows and sub-flows to generate random values. Each flow generates non-overlapping sub-flows. To execute multiple independent simulations, the seed is fixed to 9, 987. We use i as sub-flow for the $i-th$ run ($i \in \mathbb{N}^*$). Each simulation considers 10 runs, and unless otherwise noted, all the results have mean values with the confidence interval (CI) considering a level of confidence of 95%.

The data traffic is randomly generated following a uniform distribution. At each data period, a sensor node might send a message with a network load probability of C. Thus, if $C = 0$, no packet is generated. if $C = 1$, every sensor node transmits in each data period. To evaluate COPPER with different workloads, C is varied between 10% and 100% ($C = 0.1$ to $C = 1$).

Each run represents one hour of the network's operation. The data messages have 540 bytes. Want to Cooperate (WTC) messages have three bytes. NACK messages present five bytes. These numbers are similar to related work [7, 34]. We follow the configuration of the acoustic modem UNET-2 [21] for the transducer configuration. It has a 2.4 kbps data rate, the frequency is centered at 4 kHz, the modulation is BPSK, 2 kHz of bandwidth, power transmission is 138 dB. Data packet transmission and reception

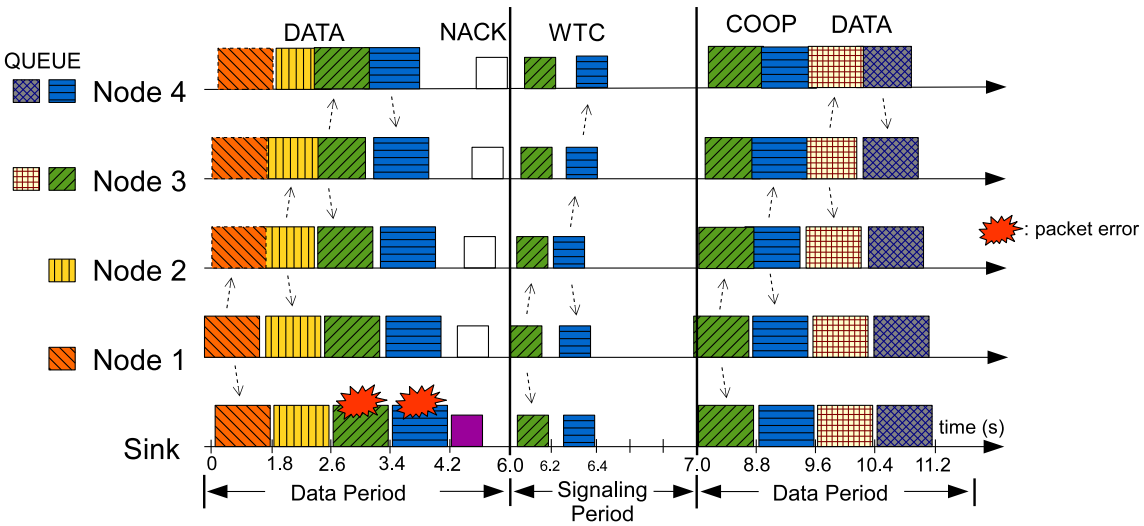


Fig. 15 Two frames of scheduled COPPER

consumes 50 W and 158 mW respectively. When idle, a radio consumes 158 mW. The data periods (DP) for every node are 2 s, where 1.8 s are for sending data and 0.2 s are for collision avoidance (guard time). The control intervals are 0.2 s for the period of signaling (SP) in synchronous COPPER. This is only 10 % of the data period. Table 2 summarizes the parameters and their values for simulating the UWSN.

Considering the parameters of Table 2 and the model we present in Sect. 3, the communication range for a node may extend for the entire network scenario. However, the Packet Error Rate (PER) varies. Figure 16 presents how the PER, for the considered scenario, varies according to the communication distance. As expected, the longer the communication distance, the lower is the probability of success to deliver a message. For example, for the worst case, where sensor and sink nodes are distant 150 m each other, (i.e., the communication range of 150 m), the probability to successfully deliver a packet is only about 5%. During our experiments, we randomly distribute the sensor nodes in the scenario. In this case, at an average distance (75 m), the probability to successfully deliver a packet directly to the sink, at the first attempt is about 45%.

Table 2 Simulation parameters

Parameters	Values
Number of independent runs	10
Time	2.000 s
Quantity of sensor nodes	50
Depth	70 m
Area	200 x 200 m
Packet payload	537 bytes
Total data packet size	540 bytes
NACK packet size	5 bytes
WTC packet size	3 bytes
Shipping factor s	0,5
Speed of wind w	2,0 m/s
Dispersion factor k	1,5
Transmission power consumption	50 W
Receiving power consumption	158 mW
Idle power consumption	158 mW
Transmission power	138 dB
Minimum SNR for reception	8
Central frequency	4 kHz
Bandwidth	2 kHz
Bits per second rate	2400
PRNG Seed	9987

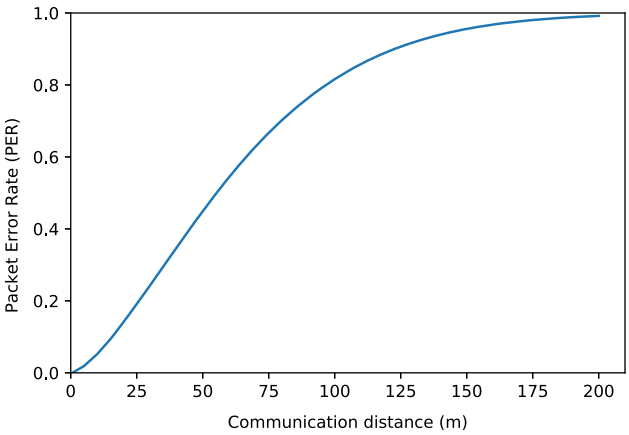


Fig. 16 Communication distance versus PER

5.2 Metrics

Throughput is one of the main MAC protocol metrics. To evaluate the protocols, we consider the throughput only for the useful data for UWSN, that is, we do not consider WTC, NACK messages or data message headers. In this way, only the payload of data messages received correctly by the sink is taken into account. This metric we call *goodput*.

Another important metric is network latency. This is the time from the input of the message in the queue of the transmitting node to the correct reception at the sink node. As pointed out by Shahabudeen et al. [66], we can increase the number of packets transmitted per time period of B and increase the throughput.

However, each time period would be so long that network latency would be degraded. In this way, it is necessary to find a middle ground between throughput and latency. In this work we consider that each node transmits one packet in its time period, that is, $L = 1$. Thus, for packets that are successfully received in the first transmission, latency will always be the lowest possible when compared to traditional TDMA.

Additionally, we calculate the Packet Error Rate (PER), which is the proportion of the total number of packets that are not received at the sink compared to the total number of transmitted packets. Moreover, we also compute the overall energy spent by the system. This is the sum of the energy spent on each sensor node in each of its operating stages: packet transmission, reception, and inactivity.

5.3 Evaluation results

First, we present the results for the execution of the four algorithms for the parameters of Table 2. We then analyze the behavior of the protocol by varying the number of nodes to obtain its scalability performance. Finally, we

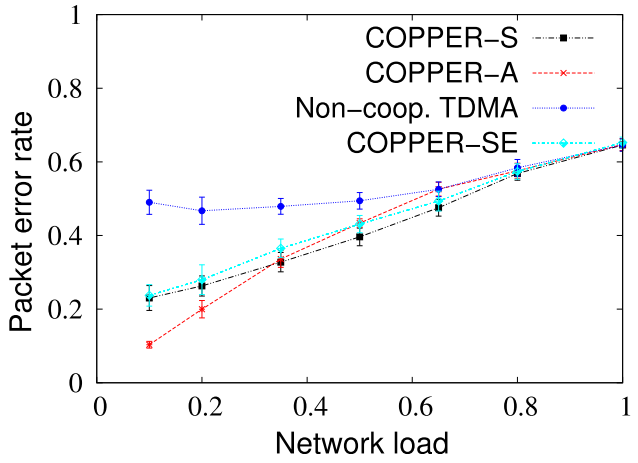


Fig. 17 Packet error rate per network load for 50 nodes

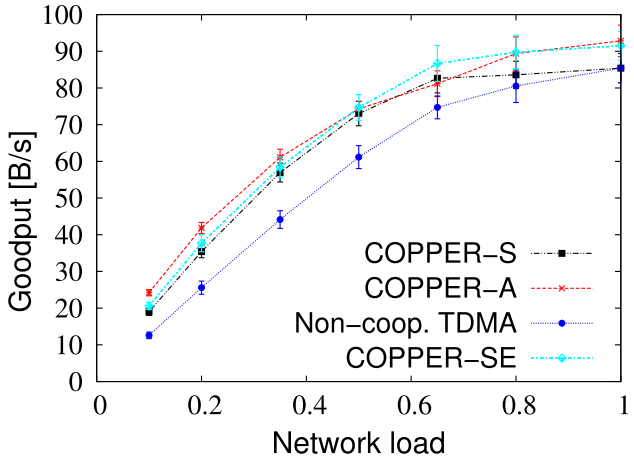


Fig. 18 Goodput per network load for 50 nodes

evaluated the results in a larger coverage area to measure the impact of high propagation delay.

Figure 17 shows the mean values of the packet error rate and the confidence interval when the network load is varied. According to this Figure, when the network load is small (i.e., <0.6) the protocols we propose presents a lower packet error rate (PER) when compared to a system that does not use any cooperative scheme. For example, the COPPER-A (Asynchronous) is almost 75% better than the UNSN using TDMA, for a 10% network load. Indeed, a TDMA UNSN presents up to 50% of packet error rate while our protocol presents about 13% only.

When analyzing the scenario with low network load (e.g., <0.3), we observed that the Asynchronous COPPER (COPPER-A) is much superior to the other protocols presenting low error rates. In this scenario, we observe a considerable number of nodes that are not transmitting and, as a consequence, they are available to cooperate. In this case, when there are a great number of available sensor nodes to cooperate, when the protocol does not synchronize the cooperation process, various nodes may retransmit a packet (i.e., the same packet). As expected, this transmission diversity increases the chance of success in data transmission. However, increasing the network workload (for instance, > 0.4) turns the COPPER-S better than the COPPER-A. The synchronization of nodes, in this case, prevents two or more nodes from retransmitting the same packet. Other nodes will transmit their own data package and, as a consequence, a higher number of distinct data will reach the sink node.

It is possible to observe a trend common to the 3 cooperative protocols in which we observe lower gains in terms of the error rate to higher network loads. In fact, in higher network loads, nodes have fewer chances to have a free time interval that they can offer for cooperation. For high network loads (e.g., > 0.8) there is almost no opportunity for cooperation and the protocols present similar results.

At lower network loads, COPPER-A also has a better flow rate (goodput) when compared to other protocols. As shown in Fig. 18, COPPER-A presents 52% better performance when compared to TDMA UWSN when the network is 10% loaded. In contrast, as occurred to the packet error rate (and for similar reasons), COPPER-S has better performance, in terms of goodput, when we increase the network load.

Despite the notable gains in terms of PER and goodput, the COPPER-A protocol requires more energy for its correct operation (Fig. 19). In fact, when several nodes attempt to cooperate without coordination or synchronism, they concurrently retransmit the same data packet. Sink node only demands a single packet and, in turn, all remaining retransmission will lead to waste energy in sensors. As shown in Fig. 19, for a low loaded network, COPPER-A consumes twice as much power as non-cooperative TDMA UWSN. In this case of low network

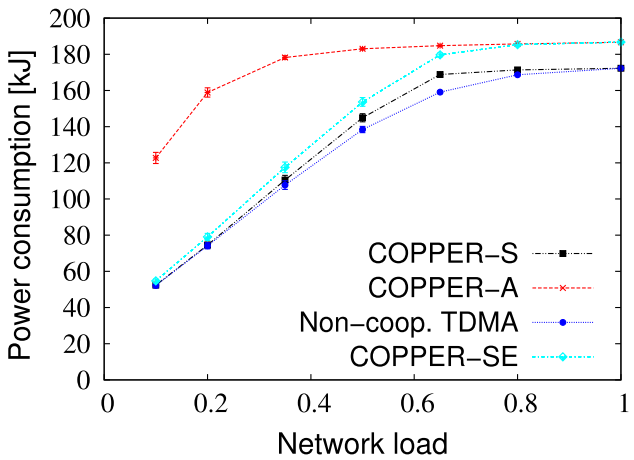


Fig. 19 Power consumption for 50 nodes per network load

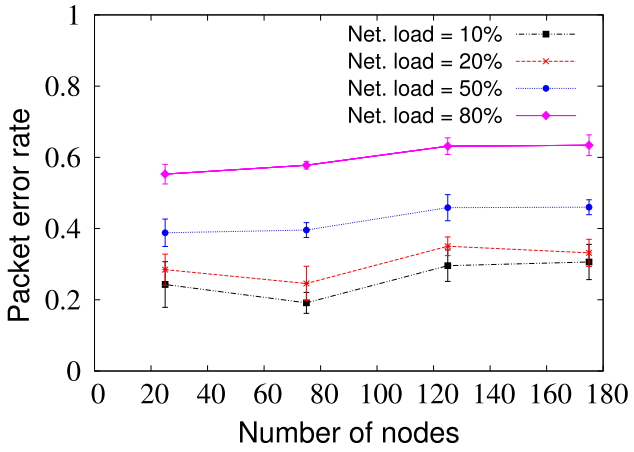


Fig. 20 COPPER-S Packet Error Rate per number of nodes with different network workloads

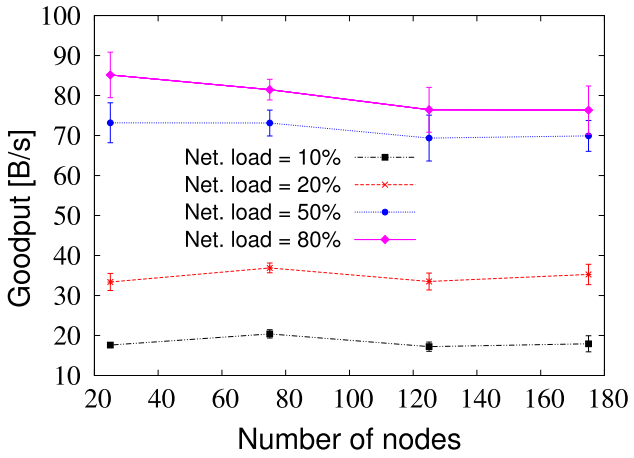


Fig. 21 COPPER-S Goodput per number of nodes with different network workloads

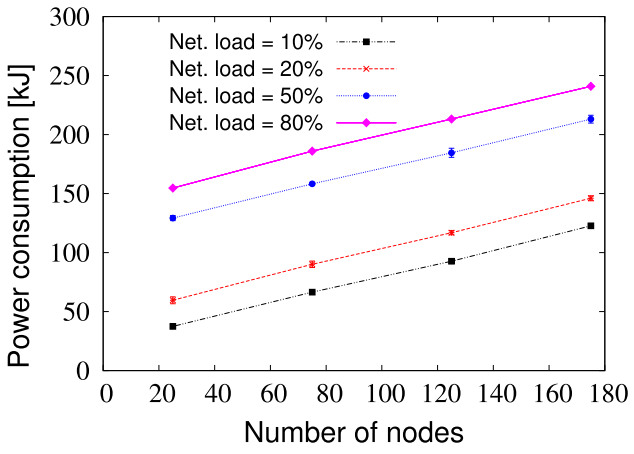


Fig. 22 COPPER-S energy consumption per number of nodes for different network loads

workload, COPPER-S and Scheduled COPPER (COPPER-SE) protocols show better performance (PER and goodput) and consume practically as much energy as the non-cooperative protocol.

Figures 20, 21 and 22 evaluate the scalability of the COPPER protocol. We focus on the COPPER-S since the synchronization leads to overhead in the protocol and, the number of system nodes affects this overhead. Figure 20 shows that the PER remains approximately stable. The goodput and the energy consumption, shown in Figs. 21 and 22, respectively, slightly increase with the network nodes number. When increasing the network size, the path diversity between communicating nodes also increases. As one could expect, path diversity can reduce the number of errors, which in turn, leads to better performance of the underwater sensor network. As the number of sensor nodes increases, it also increases the consumption of energy. In this scenario, however, the energy consumption scales. Moreover, as mentioned earlier, for higher network loads, we also observe a lower network performance. For example, at a loaded network, we observe lower goodput and higher packet error rate (PER) and power consumption.

Figure 23 shows the proportional amount of packet type sent by each protocol. In this simulation, we considered 50 nodes and a network load of 0.5. It is possible to observe that COPPER-A is the protocol with the highest index of cooperation. Again, as there is a cooperation control mechanism, a packet can be relayed by multiple nodes. The scheduled protocol, in turn, transmits fewer cooperative packets when compared to COPPER-A and COPPER-S. This happens because in advancing the transmission of a packet, it can collide with another transmission from other nodes. In this way, fewer nodes can successfully receive a copy of the transmitted packet, that is, there will be less probability of cooperation.

Next, we analyzed a simulation scenario where the nodes are in a larger area: a 1000 m side square. In this scenario, the wind speed parameter was changed to $w = 0$

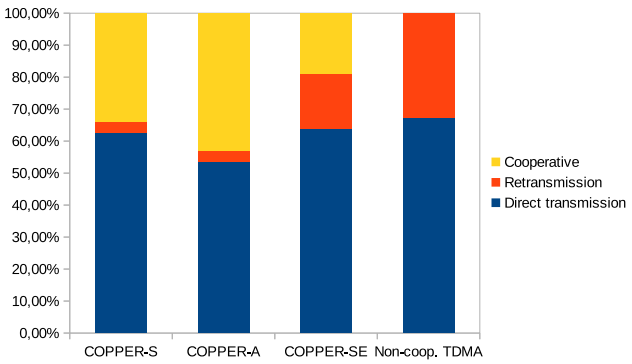


Fig. 23 Types of packets sent by the protocol

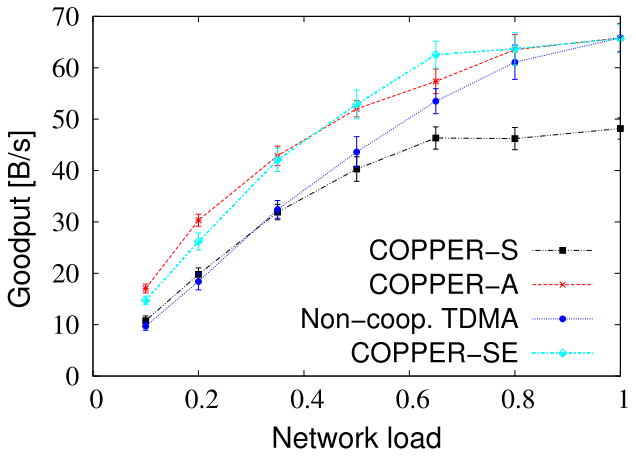


Fig. 24 Goodput per network load for a square area of size 1000 m

to obtain BER values similar to the previous scenario to evaluate the propagation delay impact.

In a larger area, the propagation delay will also be greater. This way, the guard time and duration of the signaling period should be updated accordingly, decreasing the efficiency of the protocols. Figure 24 shows the goodput of this scenario and illustrates the impact of these changes. First, when we analyze the difference between the COPPER-A and the COPPER-S, we see how the use of the signaling period in a network with a high propagation delay can degrade goodput due to channel underutilization. At the same time, when we compare COPPER-S with non-cooperative TDMA for network loads where there is greater cooperation (e.g., <0.8) and network loads with little or no co-operation (e.g., > 0.8), we observe that the gains with cooperation are significant. In the same way, the COPPER-A and COPPER-SE cooperative protocols outperform traditional TDMA in the same network load range (there is intense cooperation). COPPER-SE, in turn, by using scheduling, achieves goodput comparable to COPPER-A. Even underutilizing the channel during the signaling period, the gains with scaling are justifiable with respect to goodput.

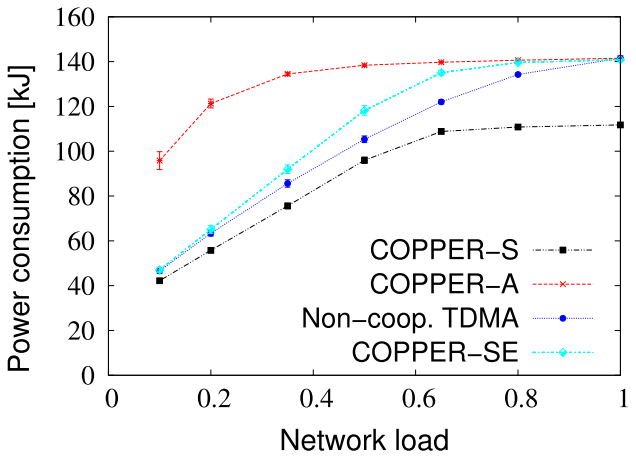


Fig. 25 Energy spent per network load for a square area of size 1000 m

Despite similar performance over goodput, the COPPER-A and COPPER-SE protocols have different energy expenditure. Figure 25 shows the expenditure of energy of the protocols in the second scenario. In this figure, we observe that the cooperative protocols that have synchronism have better energy expending than the COPPER-A. In this way, we note that for network loads <0.8 COPPER-SE has better goodput than COPPER-S and non-cooperative TDMA, and better power consumption than COPPER-A.

6 Conclusions

In this article, we have presented COPPER, a new MAC protocol for underwater wireless sensor networks that take advantage of cooperative communication. COPPER exploits the broadcast nature of the underwater environment and dynamically chooses idle sensor nodes to operate as relays. COPPER also uses selective repeat ARQ schemes to recover from errors and improve communication. We have also studied three distinct versions of COPPER: the synchronous COPPER, the asynchronous COPPER, and the data a scheduled COPPER which explores the distance between underwater nodes and the sink node to properly schedule data transmission. The data transmission scheduling reduces the transmission collision and, as a consequence, enhance the underwater data communication.

Differently from the previous works which are mostly related to the physical layer and explore amplify-and-forward mechanisms, our approaches treat cooperative communication on the MAC layer. COPPER can be used together with existing mechanisms and, in this sense, can turn the communication process even more effective. In fact, the results from our simulations evidence that COPPER improves network performance by decreasing the packet loss rate and increasing the UWSN goodput. For instance, the synchronous COPPER is able to improve the UWSN goodput by 17% and decrease the packet error rate by 65%, consuming less than 1% more energy.

Acknowledgements The authors would like to thank the research agencies CAPES, CNPq, FAPESP, and FAPEMIG.

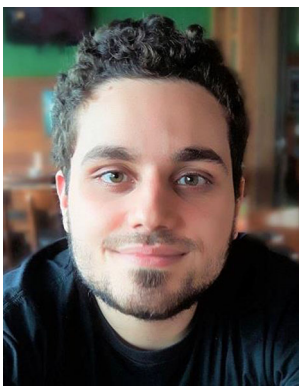
References

1. Acar, G., & Adams, A. (2006). Acmenet: An underwater acoustic sensor network protocol for real-time environmental monitoring

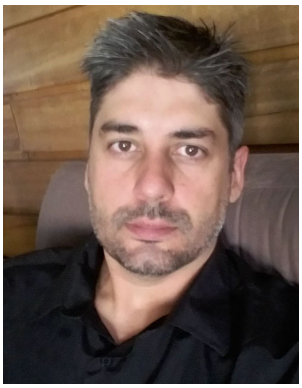
in coastal areas. *IEE Proceedings-Radar, Sonar and Navigation, 153*(4), 365–380.
2. Ahmad, A., Ahmed, S., Imran, M., Alam, M., Niaz, I. A., & Javaid, N. (2017). On energy efficiency in underwater wireless sensor networks with cooperative routing. *Annals of Telecommunications, 72*(3–4), 173–188.
3. Akyildiz, I. F., Pompili, D., & Melodia, T. (2004). Challenges for efficient communication in underwater acoustic sensor networks. *ACM Sigbed Review, 1*(2), 3–8.
4. Akyildiz, I. F., Pompili, D., & Melodia, T. (2005). Underwater acoustic sensor networks: Research challenges. *Ad hoc Networks, 3*(3), 257–279.
5. Anjangi, P., & Chitre, M. (2015). Design and implementation of super-tdma: A mac protocol exploiting large propagation delays for underwater acoustic networks. In *ACM WUWNet. ACM*.
6. Azad, S., Casari, P., Guerra, F., & Zorzi, M. (2011). On arq strategies over random access protocols in underwater acoustic networks. In *Proceedings of IEEE OCEANS*.
7. Azad, S., Casari, P., & Zorzi, M. (2013). The underwater selective repeat error control protocol for multiuser acoustic networks: Design and parameter optimization. *IEEE Transactions on Wireless Communications, 12*(10), 4866–4877.
8. Badia, L., Mastrogiovanni, M., Petrioli, C., Stefanakos, S., & Zorzi, M. (2007). An optimization framework for joint sensor deployment, link scheduling and routing in underwater sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review, 11*(4), 44–56.
9. Bainbridge, S., Eggeling, D., & Page, G. (2011). Lessons from the field-two years of deploying operational wireless sensor networks on the great barrier reef. *Sensors, 11*(7), 6842–6855.
10. Basagni, S., Petrioli, C., Petroccia, R., & Spaccini, D. (2015). Carp: A channel-aware routing protocol for underwater acoustic wireless networks. *Ad Hoc Networks, 34,* 92–104.
11. Brekhovskikh, L. M., Lysanov, Y. P., & Beyer, R. T. (1991). Fundamentals of ocean acoustics. *The Journal of the Acoustical Society of America, 90*(6), 3382–3383.
12. Burrowes, G., & Khan, J. Y. (2011). Short-range underwater acoustic communication networks. In *Autonomous underwater vehicles*. InTech.
13. Carbonelli, C., Chen, S.-H., & Mitra, U. (2009). Error propagation analysis for underwater cooperative multi-hop communications. *Ad Hoc Networks, 7*(4), 759–769.
14. Carbonelli, C., & Mitra, U. (2006). Cooperative multihop communication for underwater acoustic networks. In *ACM WUWNet*.
15. Cario, G., Casavola, A., Gjanci, P., Lupia, M., Petrioli, C., & Spaccini, D. (2017). Long lasting underwater wireless sensors network for water quality monitoring in fish farms. In *OCEANS 2017-Aberdeen* (pp. 1–6). IEEE.
16. Casavola, A. (2019). Nemo - the environmental monitoring and the automation of underwater fish farms. http://projects.dimes.unical.it/nemo/, 2017. Accessed: October.
17. Cerqueira, L. S., Vieira, A. B., Vieira, L. F., Vieira, M. A., & Nacif, J. A. (2018). Copper: Increasing underwater sensor network performance through nodes cooperation. In *2018 IEEE symposium on computers and communications (ISCC)* (pp. 00322–00327). IEEE.
18. Cerqueira, L. S., Vieira, A. B., Vieira, L. F., Vieira, M. A., & Nacif, J. A. M. (2017). Cooperative protocol for medium access control in underwater acoustic sensor networks. In *Magnetic communications: From theory to practice* (pp. 151–168). CRC Press.
19. Chitre, M., Motani, M., & Shahabudeen, S. (2012). Throughput of networks with large propagation delays. *IEEE Journal of Oceanic Engineering, 37*(4), 645–658.
20. Chitre, M., Shahabudeen, S., & Stojanovic, M. (2008). Underwater acoustic communications and networking: Recent advances and future challenges. *Marine Technology Society Journal, 42*(1), 103–116.
21. Chitre, M., Topor, I., & Koay, T.-B. (2012). The unet-2 modem–an extensible tool for underwater networking research. In *IEEE OCEANS*.
22. Coutinho, R. W., Boukerche, A., Vieira, L. F., & Loureiro, A. A. (2015). Modeling and analysis of opportunistic routing in low duty-cycle underwater sensor networks. In *ACM MSWiM*.
23. Coutinho, R. W., Boukerche, A., Vieira, L. F., & Loureiro, A. A. (2016). Geographic and opportunistic routing for underwater sensor networks. *IEEE Transactions on Computers, 65*(2), 548–561.
24. Coutinho, R. W., Vieira, L. F. M., & Loureiro, A. A. F. (2013). DCR: Depth-controlled routing protocol for underwater sensor networks. In *2013 IEEE symposium on computers and communications (ISCC)*.
25. Coutinho, R. W.L ., Boukerche, A., Vieira, L. F. M., & Loureiro, A. A. F. (2014). GEDAR: Geographic and opportunistic routing protocol with depth adjustment for mobile underwater sensor networks. In *ICC*.
26. Coutinho, R. W. L., Boukerche, A., Vieira, L. F. M., & Loureiro, A. A. F. (2016). Design guidelines for opportunistic routing in underwater networks. *IEEE Communications Magazine, 54*(2), 40–48.
27. Cui, J.-H., Kong, J., Gerla, M., & Zhou, S. (2006). The challenges of building mobile underwater wireless networks for aquatic applications. *IEEE Network, 20*(3), 12–18.
28. Demirors, E., Sklivanitis, G., Santagati, G. E., Melodia, T., & Batalama, S. N. (2014). Design of a software-defined underwater acoustic modem with real-time physical layer adaptation capabilities. In *Proceedings of the ACM international conference on underwater networks & systems*.
29. Diamant, R., Casari, P., & Zorzi, M. (2016). A TDMA-based mac protocol exploiting the near-far effect in underwater acoustic networks. In *OCEANS 2016-Shanghai* (pp. 1–5). IEEE.
30. Domingo, M. C. (2011). A distributed energy-aware routing protocol for underwater wireless sensor networks. *Wireless Personal Communications, 57*(4), 607–627.
31. Doosti-Aref, A., & Ebrahimzadeh, A. (2018). Adaptive relay selection and power allocation for OFDM cooperative underwater acoustic systems. *IEEE Transactions on Mobile Computing, 17*(1), 1–15.
32. Faustine, A., Mvuma, A. N., Mongi, H. J., Gabriel, M. C., Tenge, A. J., & Kucel, S. B. (2014). Wireless sensor networks for water quality monitoring and control within lake victoria basin: Prototype development. *Wireless Sensor Network, 6*(12), 281.
33. Felemban, E., Shaikh, F. K., Qureshi, U. M., Sheikh, A. A., & Qaisar, S. B. (2015). Underwater sensor network applications: A comprehensive survey. *International Journal of Distributed Sensor Networks, 11*(11), 896832.
34. Ghosh, A., Lee, J.-W., & Cho, H.-S. (2013). Throughput and energy efficiency of a cooperative hybrid arq protocol for underwater acoustic sensor networks. *Sensors, 13*(11), 15385–15408.
35. Gibson, J., Larraza, A., Rice, J., Smith, K., & Xie, G. (2002). On the impacts and benefits of implementing full-duplex communications links in an underwater acoustic network. Technical report, NAVAL POSTGRADUATE SCHOOL MONTEREY CA.
36. Han, J.-W., Ju, H.-J., Kim, K.-M., Chun, S.-Y., & Dho, K.-C. (2008). A study on the cooperative diversity technique with amplify and forward for underwater wireless communication. In *Proceedins of IEEE OCEANS*.
37. Han, Z., Sun, Y. L., & Shi, H. (2008). Cooperative transmission for underwater acoustic communications. In *IEEE ICC*.
38. Heidemann, J., Stojanovic, M., & Zorzi, M. (2012). Underwater sensor networks: Applications, advances and challenges.

Philosophical Transactions of the Royal Society A, 370(1958), 158–175.
39. Henry, N. F., & Henry, O. N. (2015). Wireless sensor networks based pipeline vandalisation and oil spillage monitoring and detection: Main benefits for nigeria oil and gas sectors. *The SIJ Transactions on Computer Science Engineering & its Applications (CSEA), 3*(1), 1–6.
40. Hong, L., Hong, F., Guo, Z.-W., & Yang, X. (2008). A TDMA-based mac protocol in underwater sensor networks. In *4th international conference on wireless communications, networking and mobile computing, 2008. WiCOM'08* (pp. 1–4). IEEE.
41. Jadhav, P., & Satao, R. (2016). A survey on opportunistic routing protocols for wireless sensor networks. *Procedia Computer Science, 79,* 603–609.
42. Jang, J., & Adib, F. (2019). Underwater backscatter networking. In *Proceedings of the ACM special interest group on data communication, SIGCOMM '19, New York, NY, USA. ACM* (pp. 187–199).
43. Javaid, N., Hussain, S., Ahmad, A., Imran, M., Khan, A., & Guizani, M. (2017). Region based cooperative routing in underwater wireless sensor networks. *Journal of Network and Computer Applications, 92,* 31–41.
44. Jiang, S. (2018). On reliable data transfer in underwater acoustic networks: A survey from networking perspective. *IEEE Communications Surveys & Tutorials, 20*(2), 1036–1055.
45. Khan, R. A. M., & Karl, H. (2014). Mac protocols for cooperative diversity in wireless lans and wireless sensor networks. *IEEE Communications Surveys & Tutorials, 16*(1), 46–63.
46. Kim, H.-w., & Cho, H.-S. (2016). A cooperative arq-based mac protocol for underwater wireless sensor networks. In *Proceedings of the 11th ACM international conference on underwater networks & systems*.
47. Koulakezian, A., Ohannessian, R., Denkilkian, H., Chalfoun, M., Joujou, M. K., Chehab, A., & Elhajj, I. H. (2008). Wireless sensor node for real-time thickness measurement and localization of oil spills. In *2008 IEEE/ASME international conference on advanced intelligent mechatronics* (pp. 631–636). IEEE.
48. Kredo, K., II., Djukic, P., & Mohapatra, P. (2009). Stump: Exploiting position diversity in the staggered TDMA underwater mac protocol. In *INFOCOM 2009, IEEE* (pp. 2961–2965). IEEE.
49. Kredo II, K., & Mohapatra, P. (2010). Distributed scheduling and routing in underwater wireless networks. In *Global telecommunications conference (GLOBECOM 2010), 2010 IEEE* (pp. 1–6). IEEE.
50. Lanbo, L., Shengli, Z., & Jun-Hong, C. (2008). Prospects and problems of wireless communication for underwater sensor networks. *Wireless Communications and Mobile Computing, 8*(8), 977–994.
51. Laneman, J. N., Tse, D. N., & Wornell, G. W. (2004). Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Transaction on Information theory, 50*(12), 3062–3080.
52. Lee, J. W., Cheon, J. Y., & Cho, H.-S. (2010). A cooperative arq scheme in underwater acoustic sensor networks. In *Proceedings of IEEE OCEANS*.
53. Lee, J. W., & Cho, H.-S. (2011). A cooperative arq scheme for multi-hop underwater acoustic sensor networks. In *Underwater technology (UT), 2011 IEEE symposium on and 2011 workshop on scientific use of submarine cables and related technologies (SSC)*.
54. Lee, U., Wang, P., Noh, Y., Vieira, L. F. M., Gerla, M., & Cui, J.-H. (2010). Pressure routing for underwater sensor networks. In *Proceedings of the IEEE conference on computer communications (INFOCOM)*.
55. Li, X., Liu, J., Yan, L., Han, S., & Guan, X. (2017). Relay selection in underwater acoustic cooperative networks: A contextual bandit approach. *IEEE Communications Letters, 21*(2), 382–385.
56. Lurton, X. (2002). *An introduction to underwater acoustics: Principles and applications*. Berlin: Springer Science & Business Media.
57. Menon, K. U., Divya, P., & Ramesh, M. V. (2012). Wireless sensor network for river water quality monitoring in india. In *2012 third international conference on computing, communication and networking technologies (ICCCNT'12)* (pp. 1–7). IEEE.
58. Mustafa, S., Khan, M. A., Malik, S. A., et al. (2012). Design of underwater sensor networks for water quality monitoring. *World Applied Sciences Journal, 17*(11), 1441–1444.
59. NS3. ns-3. https://www.nsnam.org/, 2018. Accessed: March, 2018.
60. Parrish, N., Tracy, L., Roy, S., Arabshahi, P., & Fox, W. L. (2008). System design considerations for undersea networks: Link and multiple access protocols. *IEEE Journal on Selected Areas in Communications*, *26*(9), 1720–1730.
61. Pasi, A. A., & Bhave, U. (2015). Flood detection system using wireless sensor network. *International Journal of Advanced Research in Computer Science and Software Engineering*, *5*(2), 386–389.
62. Pinto, D., Viana, S. S., Nacif, J. A. M., Vieira, L. F., Vieira, M. A., Vieira, A. B., & Fernandes, A. O. (2012). Hydronode: a low cost, energy efficient, multi purpose node for underwater sensor networks. In *Proceedings of the IEEE conference on local computer networks (LCN)*.
63. Rahmati, M., & Duman, T. M. (2014). Achieving delay diversity in asynchronous underwater acoustic (UWA) cooperative communication systems. *IEEE Transactions on Wireless Communications, 13*(3), 1367–1379.
64. Rappaport, T. S., et al. (1996). *Wireless communications: Principles and practice* (Vol. 2). New Jersey: Prentice hall PTR.
65. Renner, C., & Golkowski, A. J. (2016). Acoustic modem for micro AUVs: Design and practical evaluation. In *ACM WUWNet*.
66. Shahabudeen, S., Chitre, M., & Motani, M. (2011). Mac protocols that exploit propagation delay in underwater networks. In *Proceedins of IEEE OCEANS* (pp. 1–6). IEEE.
67. Sozer, E. M., Stojanovic, M., & Proakis, J. G. (2000). Underwater acoustic networks. *IEEE Journal of Oceanic Engineering, 25*(1), 72–83.
68. Stojanovic, M. (2008). Underwater acoustic communications: Design considerations on the physical layer. In *Proceedings of the fifth annual conference on wireless on demand network systems and services WONS* (pp. 1–10). IEEE.
69. Stojanovic, M., & Preisig, J. (2009). Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Communications Magazine, 47*(1), 84–89.
70. Tu, K., Duman, T.M., Proakis, J.G., & Stojanovic, M. (2010). Cooperative mimo-ofdm communications: Receiver design for doppler-distorted underwater acoustic channels. In *IEEE ASILOMAR*.
71. Tuna, G., Arkoc, O., & Gulez, K. (2013). Continuous monitoring of water quality using portable and low-cost approaches. *International Journal of Distributed Sensor Networks, 9*(6), 249598.
72. Tyan, S., Oh, S.-H., Domingo, M., & Caiti, A. (2013). Auv-rm: underwater sensor network scheme for auv based river monitoring. *Research Trend in Computer and Applications, SERCE, 24,* 53–55.
73. Urick, R. J. (1975). *Principles of underwater sound-2*. New York, NY (USA): McGraw-Hill Book.
74. Vajapeyam, M., Vedantam, S., Mitra, U., Preisig, J. C., & Stojanovic, M. (2008). Distributed space-time cooperative schemes for underwater acoustic communications. *IEEE Journal of Oceanic Engineering, 33*(4), 489–501.

75. Vieira, L., Loureiro, A., Fernandes, A., & Campos, M. (2010). Redes de sensores aquáticas. *XXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, Gramado, RS, Brasil, 1,* 199–240.
76. Vieira, L. F. M. (2012). Performance and trade-offs of opportunistic routing in underwater networks. In *IEEE WCNC*.
77. Vieira, L. F. M., Vieira, M. A. M., Pinto, D., Nacif, J. A. M., Viana, S. S., & Vieira, A. B. (2012). *HydroNode: An underwater sensor node prototype for monitoring hydroelectric reservoirs*. New York: Association for Computing Machinery. https://doi.org/10.1145/2398936.2398988.
78. Vieira, L. F. M., Vieira, M. A. M., Nacif, J. A. M., & Borges Vieira, A. (2018). Autonomous wireless lake monitoring. *Computing in Science Engineering*, *20*(1), 66–75. https://doi.org/10.1109/MCSE.2017.2581140.
79. Wang, P., Feng, W., Zhang, L., & Li, V. O. (2011). Asynchronous cooperative transmission in underwater acoustic networks. In *IEEE symposium on underwater technology*.
80. Wenz, G. M. (1962). Acoustic ambient noise in the ocean: Spectra and sources. *The Journal of the Acoustical Society of America, 34*(12), 1936–1956.
81. Wills, J., Ye, W., & Heidemann, J. (2006). Low-power acoustic modem for dense underwater sensor networks. In *Proceedings the 1st ACM international workshop on Underwater networks* (pp. 79–85). ACM.
82. Yalcuk, A., & Postalcioglu, S. (2015). Evaluation of pool water quality of trout farms by fuzzy logic: Monitoring of pool water quality for trout farms. *International Journal of Environmental Science and Technology, 12*(5), 1503–1514.
83. Yu, W., Chen, Y., Tang, Y., & Xu, X. (2018). Power allocation for underwater source nodes in uwa cooperative networks. In *2018 IEEE international conference on signal processing, communications and computing (ICSPCC)* (pp. 1–6). IEEE.
84. Zhou, Z., Yao, B., Xing, R., Shu, L., & Bu, S. (2015). E-carp: An energy efficient routing protocol for uwsns in the internet of underwater things. *IEEE Sensors Journal, PP*(99), 1.



Lucas S. Cerqueira is graduated in Information Systems from the Universidade Federal de Juiz de Fora (UFJF) in 2015 and a master's degree in Computer Science from UFJF (2018). He is currently working as Software Developer at SOTI. His interests are Algorithms, Data structures, and Computer Networks.



Alex B. Vieira is an associate professor at the Computer Science Department of Universidade Federal de Juiz de Fora. His research interests include sensor networks; IoT, network characterization, modeling, and analysis; and network science. Vieira received a Ph.D. in computer science from Universidade Federal de Minas Gerais, Brazil. Dr. Vieira has a productivity in research scholarship, awarded by the Brazilian National Council for Technological and Scientific Development (CNPq). During 2019, Vieira was a visiting professor at the National Laboratory for Scientific Computing (LNCC), Brazil. He also received several awards for distinguished research.

Luiz F. M. Vieira is graduated in Computer Science from the Universidade Federal de Minas Gerais (UFMG) in 2002, master's degree in Computer Science from UFMG (2004) and doctorate in Computer Science from the University of California Los Angeles (UCLA), United States (2009). He is currently Associate Professor at the Department of Computer Science at UFMG. Has experience in Computer Science, with emphasis on Computer Networks. He has a CNPq level 1D productivity scholarship.

Marcos A. M. Vieira holds graduated in Computer Science from the Universidade Federal de Minas Gerais (UFMG) in 2002, master's degree in Computer Science from UFMG (2004), and doctorate in Computer Science from the University of Southern California (USC), United States (2010). Post-Doctorate (2018-19) from the University of Southern California (USC), United States. He is currently Associate Professor at the Department of Computer Science at UFMG. %Has experience in Computer Science, with emphasis on Computer Networks. He is a CNPq productivity fellow.

José A. M. Nacif is a computer science professor at Universidade Federal de Viçosa, Brazil, since 2010. He holds a Ph.D. degree in computer science from Universidade Federal de Minas Gerais. He is the leader of the Computer System Engineering Laboratory at UFV and has published around a hundred papers in peer-reviewed journals and conferences. His research interests include the Internet of Things, Reconfigurable Computing, and Electronic Design Automation. Prof. Nacif is a member of IEEE, SBC, and SBMICRO.