UPMAC: A Localized Load-Adaptive MAC Protocol for Underwater Acoustic Networks

Ming Zhu, Wenzhe Zhang, Naigao Jin, Zhenquan Qin, *Member, IEEE*, Jiajun Xin, and Lei Wang

***Abstract*— Unlike terrestrial networks that mainly rely on radio waves for communications, underwater networks utilize acoustic waves, which have comparatively lower loss and longer range in underwater environments. However, acoustic waves incurs long propagation delays that typically lead to low throughput and higher energy cost of transmission than reception. Thus, collision and retransmission should be reduced in order to reduce energy cost and improve throughput. Receiver-based protocols, like receiver-initiated packet train and cluster-based on-demand time sharing, can significantly reduce collision and retransmission. But they are either time-consuming or energy consuming because nodes are controlled to turn into receiver mode by control packets or a timer regardless of load. In this paper, we propose an underwater practical MAC protocol (UPMAC), whose objective is to be adaptive to the network load conditions by providing two modes (high and low load modes) and switching between them based on different offered load. Turn-around time overhead is reduced and it is less vulnerable to control packet corruption by using the technique of piggyback. UPMAC uses receiver-based approach in high load mode, leading to a low data collision rate in both one-hop and multihop situation. Extensive simulations show that UPMAC can achieve better performance in both general and Sea Swarm topologies.**

***Index Terms*— Underwater acoustic sensor networks (UASNs), MAC protocols, receiver-initialed, load-adaptive.**

I. Introduction

UNDERWATER acoustic sensor networks (UASNs) have drawn greatly increasing attention in the past decade, since there are a wide range of applications, such as disaster prevention, exploring underwater resources, monitoring of ocean currents, etc. Due to the high loss of radio signal, acoustic waveform is widely considered as a good choice for underwater communications when compared to radio wave. However, some disadvantages inherent in acoustic waves, such as long propagation delays, low available bandwidth and high dynamic channel, have pose significant challenges and complexities to network protocol design. Low cost and gradually mature manufacturing technology have made the deployment of UASNs possible [4]. Then reliable and efficient data transmission needs to be considered.

The focus of our work is to provide high performance MAC (Medium Access Control) in UANs. Throughput and power consumption are two important metrics assessing the efficiency of MAC protocols. Collisions on which the many MAC protocols have paid attention have a significantly negative effect on the performance of UANs since they can severely reduce the throughput and increase the power cost due to the high latency and high transmission power cost [5], [6]. Over the past few decades, many kinds of MAC solutions that specifically address collision-free MAC protocols in UANs have been proposed [3], [7], [8]. In this paper, we are also eager to provide a less conflicted MAC protocol.

Besides collisions, there are many other factors influencing throughput and power consumption. Here we concern the reduction on throughput caused by the overhead of protocols, such as control packets transmission in the handshaking-based protocols, because even a short packet suffers from long transfer delay in UANs reality. In addition, a proper scheduling of nodes' data transmission needs not only to reduce collisions, but also to manage senders' transmission starting time in a pipeline way to make full use of the channel capacity.

Even though many MAC protocols have been proposed [9], [10], they are less flexible or self-adjusted since they take the same measures when encountering different situations [11]. This immobility is tolerant or reparable in the high speed radio networks. But when things change into the high delay, low speed networks, it would lead to performance reduction or sometimes even disasters when the protocols do not adjust to the network conditions. Therefore, to enhance performance in the time and space varying network, we propose a network load-adaptive MAC protocol.

UPMAC exhibits the following features: First, it is adaptive to the network load conditions by transiting between two modes: low load mode and high load mode, in which Aloha and Receiver-based protocols are used respectively. A node may have different modes on different links. Second, UPMAC provides a low data collision rate, including "transmit-receive" collision and "receive-receive" collision in both one-hop and multi-hops situation because we use Receiver-based approach in high load mode. Third, turn-around time overhead is reduced and it is less vulnerable

Manuscript received March 31, 2015; revised May 23, 2015; accepted June 8, 2015. Date of publication June 24, 2015; date of current version April 26, 2016. This work was supported in part by the Natural Science Foundation of China under Grant 61070181, Grant 61272524, and Grant 61202442, in part by the Fundamental Research Funds for the Central Universities under Grant DUT15QY05 and Grant DUT15QY51, in part by the Guangdong University of Petrochemical Technology Internal Project under Grant 2012RC0106, and in part by the Open Fund through the Guangdong Provincial Key Laboratory of Petrochemical Equipment Fault Diagnosis under Grant GDUPTKLAB201323. The associate editor coordinating the review of this paper and approving it for publication was Dr. Hongyi Wu. *(Corresponding author: Naigao Jin.)*

The authors are with the School of Software, Dalian University of Technology, Dalian 116024, China (e-mail: zhuming@dlut.edu.cn; hansonzhe@gmail.com; ngjin@dlut.edu.cn; qzq@dlut.edu.cn; brianxin1992@gmail.com; lei.wang@dlut.edu.cn).

Digital Object Identifier 10.1109/JSEN.2015.2449242

to control packet corruption, since we reduce the use of control packets by the technique of piggyback. To the best of our knowledge, no previous related schemes have used piggyback to convey the control message in underwater acoustic sensor networks. Last, UPMAC is suitable to the dynamic networks, since it provide a simple synchronization mechanism to calculate delays between nodes before transiting into the high load mode.

The Remainder of this paper is organized as follows. In Section II, we describe briefly some related work and motivations in MAC protocol design for underwater acoustic networks. We then present in Section III the UPMAC protocol. Section IV describes the simulations that were carried out to compare the performance of the proposed scheme. Finally, we give our conclusions in Section V.

II. Related Work

Previous research efforts on MAC protocols in underwater network can be generally divided into two kinds of approaches. One adopts a centralized control (scheduled-based) approach, in which each node is assigned a time period in which it is able to transmit. Zhu *et al*. [3] explored the capability of time sharing-based MAC and proposed Cluster-based On-Demand Time Sharing MAC (COD-TS) protocol. Nodes are organized into clusters, and the cluster heads are the nodes in charge of assigning the slots for the communication round. Though it can reduce collision and improve the throughput, it's not extensible when networks are dynamic, such as the cases there are several levels of the clusters or nodes uncertainly fail or join.

On the other hand, the distributed (random-based) approach, in which each node decides on its own whether to send out a packet, appears to have lower message delays and higher network utilization. Aloha-based MAC protocol is a simple and useful version of distributed approach. Chirdchoo *et al*. [5] studied the performance of Aloha-based protocols in underwater networks, and proposed two enhanced schemes, Aloha-CA (Aloha with Collision Avoidance) and Aloha-AN (Aloha with Advanced Notification). Aloha-CA tries to avoid collisions by overhearing the transmitted packets and knowing the propagation delays between all node pairs. Aloha-AN consists of sending a short data packet prior to the actual data transmission with information on the sender and the intended receiver. But it lacks the ability to overcome "receive-receive" collision [6].

Another typical kind of distributed approach adopts handshaking-based (Reservation Access) mechanism, which requires to exchange multiple small control packets prior to transmitting a longer data packet. Multiple access collision avoidance (MACA) is the first approach of a handshake algorithm proposed to reserve the channel. However, long propagation delays in underwater sensor networks makes it energy and time consuming to use RTS/CTS to eliminate the data packet collisions completely [1], [10]. There are extensive ad-hoc MAC protocols are designed based on this idea, such as MACAW [12], MACA-BI [13]. Even though several modifications have been made on the traditional MACA protocols, these handshaking-based protocols still consume too much time in terms of transmitting data.

The handshaking-based MAC protocols in UASNs evolve into two branches. One derives the idea of PCAP [14], in which PCAP makes the propagation delay predictable by allowing a node to perform other actions while waiting for the CTS frame to establish the link. Based on PACP, Guo *et al*. proposed an adaptive propagation-delay-tolerant collision-avoidance protocol (APCAP) [15]. Instead of sending CTS and DATA immediately compared to classic CSMA/CA scheme, it sets CTS window and DATA window indicating the time it is available for receiving CTS and transmitting DATA. Another one is determining the busy duration by extracting information from the packets nodes overhear [5]. M-FAMA [1] leverages passively-acquired local information (i.e., neighboring nodes' propagation delay maps and expected transmission schedules) to launch multiple simultaneous sessions. T-Lohi [16] requires nodes to start a reservation period transmitting a short control packet (tone packet) to count the number of terminals contending for the channel. If a node does not receive any other tones, it starts the transmission. Otherwise, it adapts its back-off time depending on the number of tones received.

The channel's utilization may be improved if multiple data packets in the form of a packet train can be transmitted for every set of handshake [3]. Based on this motivation, Chirdchoo *et al*. [2] proposed RIPT (Receiver-initiated Packet Train) protocol. The key difference is that the reservations are receiver-initiated. Receiver-initiated protocols have the inherent ability to address the hidden terminal issue, thus, they can dramatically reduce the collisions. However, RIPT uses another kind of 4-way handshaking to replace the tradition one, which leads RIPT has the worst delay performance in their simulations (approximately 1000s/packet with the offered load per node of 0.03). It is unacceptable with so much delay in the resource restrained environment. As will be explained in Section III later on, our approach combines the advantages of receiver-initiated protocol and the idea of packet train. The latter data transmission request is carried by the prior transmitted data. When receivers are ready to receive, it schedules the senders to send data in the way of packet train.

Reducing or eliminating packet collisions is an important issue investigated by the protocols above. Rodoplu and Park [6] and Syed *et al*. [16] investigate the reason why collisions happen in underwater acoustic network. Syed *et al*. think that the large propagation delay of acoustic media make it essential to consider the locations of a receiver and potential interferers. Distance between nodes translates into uncertainty of current global channel status: space-time uncertainty [16]. Noh *et al*. [17] also investigate the temporal and spatial reuse in underwater sensor networks. Rodoplu and Park [6] mentioned that, in a protocol, a collision can occur in two ways: First, a "transmit-receive" collision occurs in which a node is transmitting while other nodes' packets aimed at it arrives. Since the transmit power is usually much higher than the required power at which packets are received, the received packet would not decoded properly at the node, which is also called as the problem of Deafness Conditions [16]. Second, a "receive-receive" collision occurs

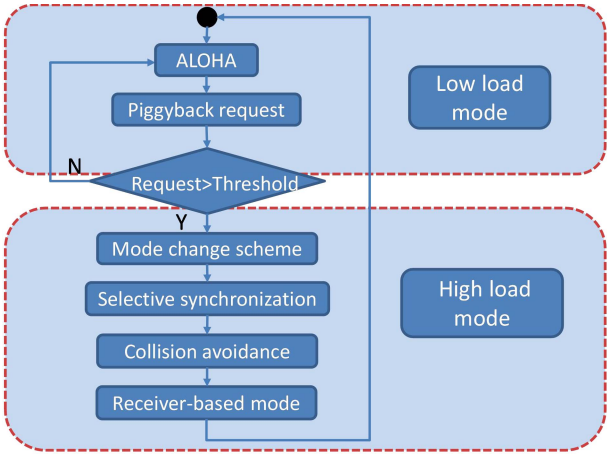

Fig. 1. UPMAC state transition diagram.

if more than two data packets arrive at a node and overlap in duration, then the node cannot decode either of the data packets. It can be seen later that our approach is immune to both of these two kinds of collisions.

III. UPMAC: Underwater Practical MAC Protocol

A. Overview

Ideally, the MAC protocols designed for underwater acoustic sensor network are supposed to be able to deal with high propagation delay, while offering high throughput and low collision rate. Moreover, they need to be adaptive to dynamic networks changes, such as density of network due to the join or leave of particular nodes and different data loads on different links. In addition, mobility is another significant feature of underwater sensor network, since nodes could float with the ocean current. It makes the delay between nodes change, which means delay is unpredictable even though after the delay detection [10]. The assumption of having acquired the information of propagation delays between all node pairs is not quite practical and even not reliable sometimes [2], [5].

In order to improve the network throughput, it is desirable to reduce the overhead of control packets while keeping a proper data transmission. Aloha-based MAC protocols are satisfied with the condition and be well investigated in [5]. It shows that Aloha-based MAC protocols perform well under low offered load. Similar results are also exhibited in [11], in which Aloha-based MAC protocols outperform some other protocols, such as T-Lohi, APCAP, slotted FAMA, and many other protocols variants under low traffic. However, when the offered load increases, the throughput reduces due to the heavy collisions encountered at the receivers. As a result, we want to investigate what would happen if the receiver could inform the senders its intention of receiving under the premise of Aloha protocol.

Our intuition is that we want convey more information upon one successful transmission. In the following, we will give more insights into UPMAC's design. UPMAC has two modes: low load mode and high load mode as showed in Fig. 1. The two modes transform with each other. In low load mode, Aloha is used for the arbitrary data transmission.

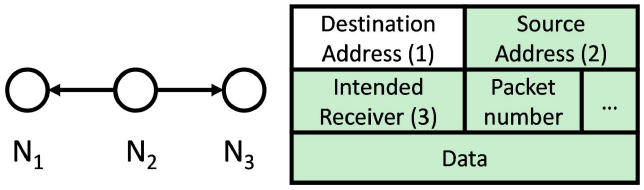

Fig. 2. An example of piggyback.

TABLE I
Notations Used for Explaining UPMAC

Symbol	Description
tx_i	node i's transmission time of one burst
d_i	delay between node i and receiver
st_i	transmission starting time of sender i in receiver's perspective
g	guard time
γ	the control packet length
τ	the maximum propagation time
δ	the data packet length
$\overline{U}$	the average amount of time in which useful data is sent during a successful busy period
$\overline{B}, \overline{I}$	the expected duration of busy and idle period
P_s	the probability of success of a packet
$S = \overline{U} / (\overline{B} + \overline{I})$	the normalized throughput
λ	the aggregate mean packet generation rate
R_n	the request number from neighbor n

The difference here is that we use the piggyback technique to inform the intended receiver that the senders want to transmit data to it after this current transmission. For example, in Fig. 2, Node 2 is transmitting data to Node 1 at present when a modified header with Intended Receiver and Intended packet size (we call them Request) are used to notify the node 3 there are some packets reserved by Node 2 for it. Node 2 won't transmit unless Node 3 let it do or the time threshold expires. Upon hearing these intending transmission information, Node 3 will estimate whether it will be busy receiving many packets with several neighbor nodes. We use a metric called *Efficiency* here to evaluate whether the node is busy. If the Requests of one node exceed the *Efficiency*, it will go into the high load mode.

In the high load mode, the receiver goes into a receiver-based mode. It first needs to calculate delays between all its direct neighbors and then, based on the Requests it heard, formulate the schedule for the senders. The idea behind the schedule is that we want to schedule different nodes' transmission time so that senders can send data in the way of pipeline. Upon hearing the schedule, senders will map this schedule into its own time line and send data at the time as the schedule specified. When the stage ends, every node turns into low load mode. The notations used in UPMAC are sumarized in Table I.

B. How the Protocol Works

As mentioned earlier, the UPMAC contains two modes, low load mode and high load mode for different load.

Algorithm 1 Went to Initiate Receiver Mode

Input:

$Eff \leftarrow \tau/\delta$ (the maximum propagation delay/average packet length); request of node N,

Begin:

```
while request > 0 do
    if request > Eff_i then
        Eff_{i+1} ← Eff_i * 2
    else
        wait for the duration that equals to the Eff_i * δ
        if Eff_i/2 > Eff then
            Eff_{i+1} ← Eff_i/2
        end if
    end if
end while
```

In addition, when altering into high load mode, the inter-node propagation delays between the Intended Receiver and each of its immediate neighbors are calculated in a short time. Therefore, UPMAC is suitable not only for the network with different traffic situation, but also for the dynamic network where nodes have the mobility or nodes can join or leave arbitrarily.

We now explain how UPMAC works. Fig. 3 shows a typical topology for illustrating how UPMAC works. In this figure the ends of the solid lines mean the sender and destination of this packet in the direction while the dotted lines mean the overhearing process (i.e., the node at the end of one dotted line is not the really destination of the packet, but it can hear and receive it).

1) Low Load (Aloha-Based) Mode: After the network is deployed, nodes send beacons to announce their existence, but they do not know the delays among each other. Nodes transmit data in an arbitrary way. Besides containing the direct receiver, when nodes transmit data, the packet header is modified by adding two fields: Intended Receiver (IR) and Intended Packet Size (IPS) as the Request for the possible transmission. Upon hearing the header, Intended Receiver will extract this information and determine whether it will be busy for a period of time in the future by calculating Request based on IR and IPS. For the direct receiver, an instantaneous ACK is returned if it receive the packet successfully.

2) High Load (Receiver-Based) Mode: As mentioned before, IR and IPS are carried along with the data transmission. Intended receiver will calculate the Requests based on all the IRs and IPSs from neighbors. If the Requests exceed the threshold, called *Efficiency* (i.e., *NR* in Equation (11)), the node turns into high load mode. Aloha can perform well at the low load mode, however, with the traffic increases, Aloha's performance will decrease due to many collisions and retransmissions.

In order to keep fair, avoiding the same Node N from acting as a receiver successively before other neighboring nodes have a chance at playing the role, we double the Node N's *Efficiency* after it acts as a receiver. If Requests of Node N is still larger than *Efficiency*, it can go into receiver mode. Otherwise, it will wait for a period which is the duration Node N will keep if it were transmitting data in receiver mode (Algorithm 1).

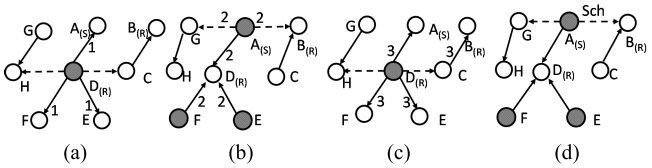

Fig. 3. Synchronization for locally estimating delays between nodes. (a) Step 1. (b) Step 2. (c) Step 3. (d) Step 4.

In order to reduce collisions, it's reasonable to ask receivers to provide more information to guide senders to send data in a scheduled way. Before making schedules, the receiver should collect enough Requests which have been done at the previous transmission by using piggyback. Connected by these Requests, these nodes formulate a virtual cluster (VC), in which the receiver is the header and the senders are the cluster members. It is not a real cluster because the senders are only part of the receiver's direct neighbor and it is temporary formed and will be dismissed after scheduled data transmissions are completed.

Before making schedules, it is essential to estimate delays between nodes. Here, we use a selective synchronization (SS) strategy as follows. If an IR is ready to receive (*Efficiency* exceeds threshold), it will calculate delays with all its neighbors within a three-way handshaking process. For the senders, their data transmission schedules are also supposed to be broadcast to their neighbors, because if some of these neighbors want to be IRs, it is crucial to avoid its interference.

For example, in Fig. 3, Node D is going to be the IR, and there are three senders, Node A, E, and F (we call these node Intended Senders, ISs). The SS process do as follows.

1) Node D first broadcast the initiated packet (ANN) to all its neighbors (Node A, C, E, F, G, and H, but only Node A, E, and F are the senders to Node D).
2) Once hearing this packet, they response packet 2 immediately to Node D so that it knows the delays with Node A, E, and F, and then calculates schedule for them. We believe Node D can hear the responses with few collisions due to the Spatial Uncertainty (mentioned in Section II). If collisions happen, Node D will ignore those nodes involved in the collisions and only calculate schedule for these nodes whose responses are received clearly. At the same time, Node B receives packet 2 from Node A.
3) Node D broadcasts an announcement packet (SCH, packet 3 in Fig. 3) containing the schedule, thus Node A, E, and F can know the delays with Node D, in the meantime, map schedule into their own time lines. Node B responses to Node A with packet 3 but Node G won't because Node B is going to be IR so Node A's transmission may make influences on its reception, while it won't affect Node G since it's a sender.
4) Node A broadcasts its own transmission schedule in its first data transmission slot. After hearing this packet, Node B knows the occupied slot caused by Node A

and will design its own reception schedule on the free slots.

There are some discussions about this process. First, for nodes like Node B and G, upon hearing Node A's packet 2 (a sender involved in another transmission), they should decide whether to synchronize based on their own roles. If they want to be IRs, they should synchronize, while ISs do not need to. If Node A is sending packet 1, then all the neighbors should synchronize with Node A no matter what kinds of roles the neighbors are. In a word, if the synchronization is initialed by a receiver (like Node D), then all the neighbors need to synchronize, while if it is initialized by a sender (like Node A), then the neighbors will selectively synchronize based on their own roles. Second, if Node B only receive Node A's packet 2 but not packet schedule, it can't go into the receiver-based mode since it doesn't know when Node A will send data. Nevertheless, if Node B only receive Node A's packet schedule but not packet 2, it can change into receiver-based mode only after it hear Node A's first data packet because it can use the offset to calculate when the rest packets come.

In this paper, data are transmitted from each sender in a burst in order to reduce the guard time of per packet, and the receiver acknowledges per schedule in order to reduce the control packet overhead and improve the channel utilization. We refer to this technique as the schedule-based acknowledgement here.

Acquiring delays to each sender in process 2, receiver could calculate schedule, transmission starting time, as the rules followed (Fig. 4). Receiver first sort these delays in descending order. The sender with shortest delay transmits its burst immediately upon receiving ANN (S_1 in Fig. 4). The sender i decides when to transmit by determining whether $2(d_i - d_{i-1})$ is larger than the sum of ith sender's data transmission time plus one guard time Equation (1). That is, if $2(d_i - d_{i-1})$ is larger than $tx_{i-1} + g$, then sender i sends data upon receiving ANN (S_2 in Fig. 4), otherwise, it won't transmit until $tx_{i-1} + g - 2$ $(d_i - d_{i-1})$ (S_3 in Fig. 4 will wait t_{defer} to transmit).

$$st_i = \begin{cases} d_i, & 2(d_i - d_{i-1}) > tx_{i-1} + g; \\ d_i + tx_{i-1} + g - 2(d_i - d_{i-1}), & \text{otherwise.} \end{cases} \tag{1}$$

C. Mode Switching Scheme and Throughput Analysis

In this section we discuss when UPMAC switches its mode and analyzes the through throughput. First, we characterize the traffic source and its underlying assumptions.

We assume that our traffic source consists of an infinite number of users [18], thus we model traffic as independent Possion arrival process with an aggregate mean packet generation rate of λ packets/s. Each node has a single packet transmission buffer with the size of B_{max}.

Assume that no other packets are transmitted in a time interval τ, the probability P_s of a packet is successfully transmitted is:

$$P_s = e^{-\tau\gamma}. \tag{2}$$

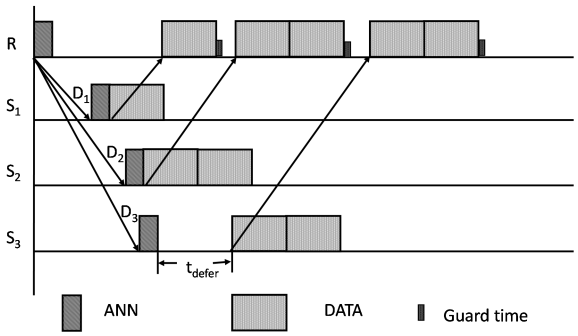

Fig. 4. Data transmitting based on schedules.

Then the successfully received requesting packet number from neighbor n is

$$N_{nRs} = R_n P_s, \tag{3}$$

where R_n is a constant and it is less than B_{max}.

A successful transmission consists of an ANN (γ) with one propagation delay (τ), a three-way synchronization ($3(\gamma + \tau)$) and a train of data packets. The consuming time T_s of a successful transmission is:

$$T_s = 2\gamma + 4\tau + NR, \tag{4}$$

where n is the id in the neighbor set N, $NR = \sum_{n \in N} \delta N_{nRs}$ is the scheduled packets number when the receiver switches its mode to receiver-based mode.

Both the loss of ANN and SCH packets can lead to an unsuccessful transmission. In the first case, an unsuccessful transmission consists of an ANN and one or more ANNs within a time Y ($0 \leq Y \leq \tau$) followed by a propagation delay [18], while in the second case, an unsuccessful transmission consists of an ANN, a propagation delay, neighbors' responses, and one or more SCH within Y. Thus, the time taken when ANN and SCH fail T_{ANNf} and T_{SCHf} can be shown:

$$T_{ANNf} = \gamma + \tau + Y = \gamma + 2\tau - \frac{1 - e^{-\tau\lambda}}{\lambda}, \tag{5}$$

$$\begin{aligned} T_{SCHf} &= \gamma + \tau + \tau + \tau + \gamma + Y \\ &= 2\gamma + 4\tau - \frac{1 - e^{-\tau\lambda}}{\lambda}. \end{aligned} \tag{6}$$

The average busy period $\bar{B}$ is expressed by:

$$\begin{aligned} \bar{B} &= T_s P_s + (T_{ANNf} + T_{SCHf})(1 - P_s) \\ &= -2/\lambda + 4e^{-\tau\lambda}/\lambda - 2e^{-2\tau\lambda}/\lambda \\ &\quad + 3(\gamma + 2\tau) - (\gamma + 2\tau)e^{-\tau\lambda} + NR\delta e^{-\tau\lambda}, \end{aligned} \tag{7}$$

The average utilization $\bar{U}$ is:

$$\bar{U} = \delta P_s NR. \tag{8}$$

Now when wanting to achieve a maximum throughput, we assume that our algorigthm were able to perfectly schedule the packets into the available channel space with absolutely no overlap or gaps between the packets. Thus the average idle

period $\bar{I} = 0$. Finally, the normalized throughput $S_{Receiver}$ is given by:

$$S_{Receiver} = \frac{\delta NR}{-\frac{-2(1-e^{-\tau\lambda})^2}{\lambda e^{-\tau\lambda}} + (\gamma + 2\tau)(3e^{\tau\lambda} - 1) + NR\delta}. \quad (9)$$

When $S_{Receiver}$ is greater than S_{ALOHA}, receiver changes its mode from ALOHA mode into Receiver-based mode. Since $S_{ALOHA} = e^{-\tau\lambda}$, we have

$$S_{Receiver} > e^{-\tau\lambda}. \quad (10)$$

From Equation (9) and Equation (10), the receiver switches into receiver-based mode when

$$NR > (\frac{-2(1-e^{-\tau\lambda})}{\lambda} + \frac{(\lambda + 3\tau)(3e\tau\lambda - 1)}{e^{\tau\lambda} - 1})\frac{1}{\delta}, \quad (11)$$

where *NR* is the *Efficiency* mentioned in Section III-B2.

D. Collision Avoidance During Packet 2 Transmission

The packet 1 sent by IR contains contention window field to indicate the period during which the IR waits for packet 2. Senders who want to send packets should reply with packet 2 within this period. If multiple nodes answer simultaneously, collision happens and colliding packet 2 is lost.

In order to reduce the collision probability, the contention window is split into a number of small time slots. Each slot equals the time to transmit a packet 2 by the sender plus the time for the IR to process the packet 2. While it is desirable to have each sender reply packet 2 in a unique time slot, however, it is not realistic since the nodes are not synchronized at this moment. An intuitive approach is to let a sender randomly select a time slot. In this simple approach, the length of contention window, i.e., W, is a crucial parameter. More senders need a larger contention window.

In the following discussion, we will propose two simple schemes to determine the optimal value of the contention window W. The former scheme considers the overall collision probability, while the latter focuses on the collision probability for those senders who have more data to send.

1) Minimizing Overall Collision Probability: In this scheme, we intend to minimize the probability of packet 2 experiencing collisions. Assume node i has sent packet 1 successfully to a set of its neighbors, denoted by $\boldsymbol{\phi}_i$. Thus the overall collision probability equals $P_o = 1 - \left[\frac{W}{|\boldsymbol{\varphi}_i|}\right] \times (|\boldsymbol{\varphi}_i|!) \times \left[\frac{1}{W}\right]^{|\boldsymbol{\varphi}_i|}$ [22]. A minimum W can be chosen by linear search, to ensure that P_o is lower than an expected value.

2) Minimizing Collision Probability for Nodes Who Want Send More Data: Since it can improve the throughput and reduce energy cost if the nodes who want to send more data can send successfully, it is reasonable to make their collision probabilities as small as possible. With this consideration, we design a collision-avoiding scheme, as elaborated below.

When a node i transmit packet 1, the following information is embedded in the packet: contention window W and total packets number N_i that it is planning to receive. If a neighboring IS j receives this packet 1, it needs to determine a time slot for transmitting its packet 2, which is a random variable between 0 and t_j, based on its own packet number n_j.

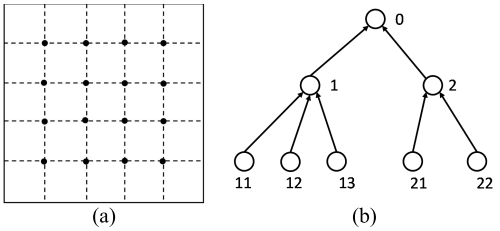

Fig. 5. Simulation network topologies. (a) Grid network. (b) Tree network.

Obviously, the larger t_j is, the less likelihood that node j will conflict with other nodes. t_j is calculated according to the following equation:

$$t_j = \frac{N_i - n_j}{N_i} \times W. \quad (12)$$

Thus, we can calculate the collision probability of one specific node as [22, eq. (16)].

IV. Simulation and Evaluation

A. Simulation Setup

To assess the performance of a dynamic network consisting of autonomous underwater vehicles (AUVs), a simulation analysis has been carried out using the Aqua-Sim [19], which can effectively simulates the attenuation of underwater acoustic channels and the collision behaviors in long delay acoustic networks. Aqua-Sim adopts the signal attenuation model as $A(l, f) = l^k \times (10^{\frac{\alpha(f)}{10}})^l$, where $A(l, f)$ is the average signal attenuation through the distance l at frequency f; k is the spread factor, which is determined by the spreading types of the acoustic signals; $\alpha(f)$ is the absorption coefficient, which can be empirically expressed by the Thorps equation [20]. Unless specified mentioned, the bit rate is set to 10 *kbps*. We set the size of control packets to 4 *bytes* and the data packet size to 60 *bytes*. Note that at the bit rate of 10 *kbps*, a 60 *bytes* frame requires less than 0.05 *sec* to transmit and the one-way trip delay on a 1500m link is about 1s, which means *Efficiency* (the ratio of the maximum propagation time to the data packet length) is about 20 [21]. We measure the throughput, delay, energy cost and collision of the total network as the functions of the offered load on the sensor network. The load is varied between generating a single frame every 100*s* down to a single frame every 2*s*. In our simulation, each run lasts for 1 hour. We report the average value of 50 runs with the 95% confidence interval.

We compare the performance of UPMAC with other three protocols: ALOHA, RIPT ([2]), and COD-TS ([3]). RIPT is a receiver-based protocol which utilizes a receiver-initiated 4-way (RTR/SIZE/ORDER/DATA) handshake. COD-TS is a cluster-based protocol in which all nodes run a distributed clustering algorithm to know when to request for data transmission. Both RIPT and COD-TS assume that the propagation delay between each pair of node is known.

B. Network Topology

As shown in Fig. 5, we deployed the nodes in a grid topology and a tree topology in a 3D region. In the grid

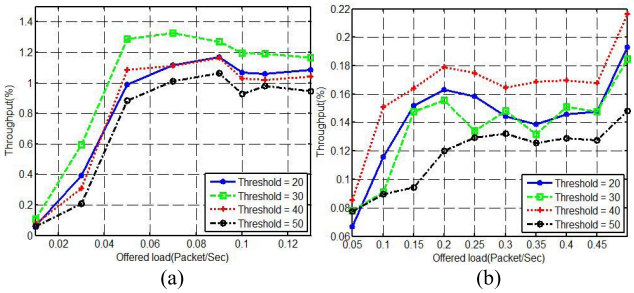

Fig. 6. Initial values of *Efficiency* for different topologies. (a) Grid network. (b) Tree network.

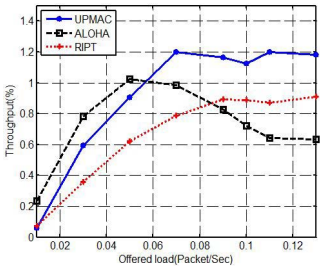

Fig. 7. Comparing the throughput of UPMAC, RIPT, and ALOHA in the grid topology.

topology depicted in Fig. 5 (a), 16 nodes are deployed in a grid with a fixed distance of 1000m between one-hop neighbors and transmission range of these nodes are 1100m. Nodes move at a constant velocity of 5 knots (2.5 m/s). We adopt this grid topology to show how spatial reuse affects system throughput and collision situation in multi-hop network. The tree topology depicted in Fig. 5 (b) is representation of a Sea Swarm engaged in monitoring an underwater region and reporting multimedia information to the servers on land. It is a unidirectional topology, which delivers a more aggressive traffic toward the sink node since the nodes in lower level attempt to simultaneously send their data to the nodes in upper level.

C. Simulation Results

Before we explore UPMAC's performance, we need to choose an appropriate initial value of *Efficiency* for different topologies. In Fig. 6, we show the throughput with different *Efficiencies*. The *Efficiency* is initially chosen as 20 and increased by half of *Efficiency*. It can be seen that the appropriate initial values of *Efficiency* for grid and tree topologies are 30 and 40 respectively. Actually, this evaluation in real experiment is not necessary because the *Efficiency* become stable in Algorithm 1 and we will use these values in our following simulation.

1) Throughput: The simulation duration for each data point was 3600s, and we adopt the definition of "Throughput per node" as the average throughput over N nodes as follows:

The throughput of UPMAC, RIPT, COD-TS, and ALOHA in different topology are shown in Fig. 7 and Fig. 8, respectively. The throughput of schedule based protocols, like RIPT, COD-TS, and UPMAC, are lower than contention

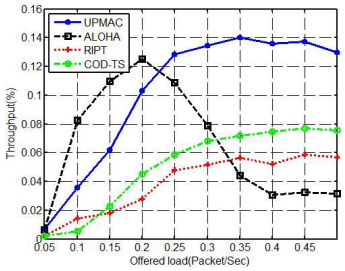

Fig. 8. Comparing the throughput of UPMAC, RIPT, COD-TS, and ALOHA in the tree topology.

based protocol, like ALOHA. The reason is that when an IR goes into receiver mode, the data transmission of neighbors of which are suppressed due to avoiding the IR's receiving packets train. With the increasing of the offered load, the throughput of schedule-based protocols rises, while the throughput of ALOHA decreases. In schedule-based protocols, data transmission are precisely scheduled in order to avoid collisions including hidden and exposed terminal collisions.

In terms of the schedule based protocols, the throughput T of each node is defined as:

$$T = \frac{1}{num}\frac{nrp^{*}ptt}{st}, \tag{13}$$

where num is the node number in simulation, nrp is the number of received packets, ptt is the transmission time of one packet, and st is simulation time. UPMAC increases faster with the rising of offer load than that of RIPT and COD-TS and keeps in a higher throughput level. In tree topology, UPMAC outperforms RIPT by 2 times and COD-TS by 60% when offer load exceed 0.25. Because COD-TS and RIPT require that nodes pre-know the propagation delays with neighbors, as a result, collisions will occur at the beginning or end of data transmission when nodes move. In addition, cluster headers in COD-TS need to avoid schedule update conflict intervals when it is extended to multi-hop networks, which will further decrease the throughput. On the other hand, RIPT stipulates that nodes should wait a threshold time or serve as a role of sender in order to avoid the same node from acting as a receiver successively. While, in UPMAC, fewer control packets are used since requests are piggybacked by data transmitted before and delay estimation before going into receiver mode ensures that data are successfully received in a chain even when nodes are moving.

2) Residual Energy: The residual energy performance of these protocols in grid and tree topology are shown in Fig. 9 and Fig. 10. The initial energy of each node in tree and grid topology is 200 mWhr and 1000 $mWhr$ respectively. At first, residual energy of schedule based protocols are higher than ALOHA at low offered load, because these protocols' throughput are low at low data rate (Fig. 8). Different with schedule based protocols, in Fig. 10, the residual energy of ALOHA does not decline, but go up a little with the increase of offered load. This is because, with the increase of data rate, data transmission of ALOHA will collide more and more nodes stays in the stage of waiting for ACK. In another word, using ALOHA most nodes will wait the responses

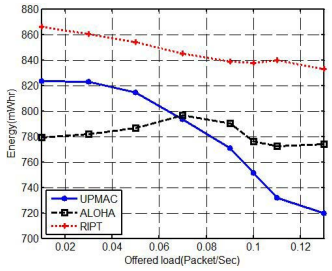

Fig. 9. Comparing the residual energy of UPMAC, RIPT, and ALOHA in the grid topology.

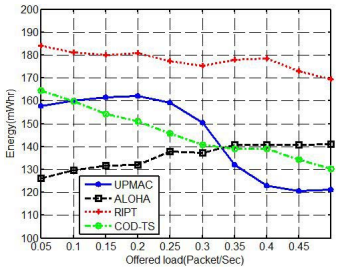

Fig. 10. Comparing the residual energy of UPMAC, RIPT, COD-TS, and ALOHA in the tree topology.

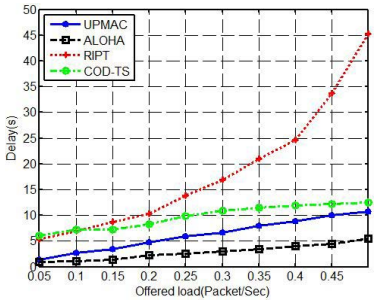

Fig. 11. Comparing the delays of UPMAC, RIPT, COD-TS, and ALOHA in the tree topology.

consuming a relative lower energy. However, in Fig. 9, when offered load is high the energy consumption also increase since the links are more intensive in grid topology than tree topology. Schedule based protocols will consume more energy when offered load is high since more data are transmitted, among which the residual energy drops rapidly because the throughput of UPMAC is high and more data are transmitted in high data rate.

3) End-to-End Delay: This metric is defined as the average time between the packet generation time and the time of its correct delivery at the destination. The average delays of these four MAC protocols in grid topology and tree topology are similar. Fig. 11 shows the delay performance of the four protocols in tree topology. When the load is low (such as 0.05), the performance of RIPT and COD-TS are worse than ALOHA and UPMAC since they are pure receiver-initialed approach, whereby a sender need wait until there is a hand-shake initiated by the receiver before it can attempt to transmit a data packet. When the offered load goes up, UPMAC is getting close to COD-TS, because more data are transmitted by receiver-based mode in UPMAC. But COD-TS need avoid schedule update conflict intervals, as a result, its delay is higher than UPMAC even when data rate is high. On the other hand, RIPT gets the worse delay performance, since the data transmission are severely dominated by receiver's willingness to be a receiver.

V. Conclusion

We have proposed a novel MAC protocol called UPMAC, which is adaptive to the network load conditions by providing two modes and using different protocols in different modes. UPMAC provides a low data collision rate both in one-hop and multi-hop situation by using Receiver-based approach in high load mode. With the less use of control packets by the technique of piggyback, turn-around time overhead is reduced and it is less vulnerable to control packet corruption, UPMAC is suitable to the dynamic networks, since it provides a simple synchronization mechanism to calculate delays between the nodes before transiting into the high load mode. Extensive simulations show that our approach can achieve significantly better performance in both general and Sea Swarm topologies especially when offered load is high.

References

[1] S. Han, Y. Noh, U. Lee, and M. Gerla, "M-FAMA: A multi-session MAC protocol for reliable underwater acoustic streams," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 665–673.

[2] N. Chirdchoo, W. S. Soh, and K. Chua, "RIPT: A receiver-initiated reservation-based protocol for underwater acoustic networks," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 9, pp. 1744–1753, Dec. 2008.

[3] Y. Zhu, Z. Jiang, Z. Peng, M. Zuba, J.-H. Cui, and H. Chen, "Toward practical MAC design for underwater acoustic networks," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 683–691.

[4] S. N. Le, Y. Zhu, and J.-H. Cui, "Pipelined transmission MAC for string underwater acoustic networks," Dept. Comput. Sci. Eng., Univ. Connecticut, Storrs, CT, USA, Tech. Rep. UbiNet-TR13-01, 2012.

[5] N. Chirdchoo, W.-S. Soh, and K.-C. Chua, "Aloha-based MAC protocols with collision avoidance for underwater acoustic networks," in *Proc. INFOCOM*, May 2007, pp. 2271–2275.

[6] V. Rodoplu and M. K. Park, "An energy-efficient MAC protocol for underwater wireless acoustic networks," in *Proc. MTS/IEEE OCEANS*, Sep. 2005, pp. 1198–1203.

[7] M. Molins and M. Stojanovic, "Slotted FAMA: A MAC protocol for underwater acoustic networks," in *Proc. OCEANS-Asia Pacific*, May 2007, pp. 1–7.

[8] Y.-D. Chen, C.-Y. Lien, Y.-S. Fang, and K.-P. Shih, "TLPC: A two-level power control MAC protocol for collision avoidance in underwater acoustic networks," in *Proc. MTS/IEEE OCEANS*, Bergen, Norway, Jun. 2013, pp. 1–6.

[9] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proc. ACM SenSys*, 2003, pp. 171–180.

[10] P. Xie and J.-H. Cui, "R-MAC: An energy-efficient MAC protocol for underwater sensor networks," in *Proc. WASA*, Aug. 2007, pp. 187–198.

[11] C. Petrioli, R. Petroccia, and M. Stojanovic, "A comparative performance evaluation of MAC protocols for underwater sensor networks," in *Proc. OCEANS*, Sep. 2008, pp. 1–10.

[12] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LAN's," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 4, pp. 212–225, Oct. 1994.

[13] F. Talucci, M. Gerla, and L. Fratta, "MACA-BI (MACA by invitation)—A receiver oriented access protocol for wireless multihop networks," in *Proc. PIMRC*, Sep. 1997, pp. 435–439.

[14] X. Guo, M. R. Frater, and M. J. Ryan, "A propagation-delay-tolerant collision avoidance protocol for underwater acoustic sensor networks," in *Proc. OCEANS-Asia Pacific*, May 2007, pp. 1–6.

[15] X. Guo, M. R. Frater, and M. J. Ryan, "Design of a propagation-delay-tolerant MAC protocol for underwater acoustic sensor networks," *IEEE J. Ocean. Eng.*, vol. 34, no. 2, pp. 170–180, Apr. 2009.

[16] A. A. Syed, W. Ye, and J. Heidemann, "T-Lohi: A new class of MAC protocols for underwater acoustic sensor networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 789–797.
[17] Y. Noh, P. Wang, U. Lee, D. Torres, and M. Gerla, "DOTS: A propagation Delay-aware opportunistic MAC protocol for underwater sensor networks," in *Proc. ICNP*, Oct. 2010, pp. 183–192.
[18] L. Kleinrock and F. A. Tobagi, "Packet switching in radio channels: Part I—Carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Trans. Commun.*, vol. 23, no. 12, pp. 1400–1416, Dec. 1975.
[19] *Aqua-Sim*. [Online]. Available: http://obinet.engr.uconn.edu/wiki/index.php/Aqua-Sim, accessed 2014.
[20] L. M. Brekhovskikh and Y. P. Lysanov, *Fundamentals of Ocean Acoustics*. New York, NY, USA: Springer, 2003.
[21] S. Keshav, *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network*. Reading, MA, USA: Addison-Wesley, 1997.
[22] Y. Wang, H. Wu, and N.-F. Tzeng, "Cross-layer protocol design and optimization for delay/fault-tolerant mobile sensor networks (DFT-MSN's)," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 5, pp. 809–819, Jun. 2008.

Ming Zhu received the bachelor's and master's degrees in communication and information system from the School of Software, Dalian University of Technology, Dalian, China, where he is currently pursuing the Ph.D. degree. His research interests include wireless sensor networks and wireless networks communications.

Wenzhe Zhang was born in 1990. He received the bachelor's degree in software engineering from the Dalian University of Technology, China, in 2012, where he is currently pursuing the master's degree. His research interests include wireless network and underwater sensor networks.

Naigao Jin received the M.S. and Ph.D. degrees from the Dalian University of Technology, China, in 2002 and 2008, respectively. He is currently a Lecturer with the School of Software, Dalian University of Technology. His research interests include sensor data fusion, localization, and tracking in sensor network.

Zhenquan Qin (M'12) received the B.S. degree in security engineering and the Ph.D. degree from the University of Science and Technology of China, in 2002 and 2007, respectively. He is currently an Associate Professor with the School of Software, Dalian University of Technology. He is a member of the China Computer Federation. His research interests include wireless network, network analysis, and network security.

Jiajun Xin was born in 1992. He is currently pursuing the bachelor's degree in network engineering from the Dalian University of Technology, Dalian, China. His research interests include wireless network and cyber physical system.

Lei Wang received the B.S., M.S., and Ph.D. degrees from Tianjin University, China, in 1995, 1998, and 2001, respectively. He is currently a Full Professor with the School of Software, Dalian University of Technology, China. He was a member of the Technical Staff with Bell Labs Research China from 2001 to 2004, a Senior Researcher with Samsung, Korea, from 2004 to 2006, a Research Scientist with Seoul National University from 2006 to 2007, and a Research Associate with Washington State University, Vancouver, WA, USA, from 2007 to 2008. His research interests involve wireless ad hoc network, sensor network, social network, and network security.