Aqua-Sim: An NS-2 Based Simulator for Underwater Sensor Networks

Peng Xie, Zhong Zhou, Zheng Peng, Hai Yan, Tiansi Hu,
Jun-Hong Cui, Zhijie Shi, Yunsi Fei, Shengli Zhou
Underwater Sensor Networks Lab
University of Connecticut, Storrs, CT 06269-2155

***Abstract*— In this paper, we present a network simulator, Aqua-Sim, for underwater sensor networks. Aqua-Sim is based on NS-2, one of the most widely used network simulators, and it follows object-oriented design style and all network entities are implemented as classes. Aqua-Sim effectively simulates the attenuation of underwater acoustic channels and the collision behaviors in long delay acoustic networks. Moreover, Aqua-Sim supports three-dimensional network deployment and provides a rich set of basic and advanced protocols. Through several case studies, we show that Aqua-Sim can "reproduce" the real world with high fidelity and flexibility.**

I. Introduction

Recently underwater sensor network has emerged as a powerful technique for aquatic applications, and it has attracted more and more attention from the networking research community [19], [13], [1], [6], [4]. To facilitate the research in underwater sensor networks, it is desirable to have a standard simulation platform to compare and evaluate different network designs, algorithms and protocols.

In the literature, there are a couple of efforts on incorporating acoustic propagation models into the simulation of underwater acoustic networks [14], [18]. To the best of our knowledge, however, there is no complete packet level underwater sensor network simulator published yet. There are several widely used packet level network simulators such as NS-2 [9] and OPNET [10]. But they were developed for terrestrial radio wireless and/or wired networks, not for underwater sensor networks. They can not be used for the simulation of underwater sensor networks without significant modifications for the following reasons: Firstly, acoustic communication is the commonly accepted method for underwater environments, while the propagation speed of acoustic signal in water is very slow (about 1500 m/s), significantly different from that of radio signal; secondly, the acoustic signal attenuation model is drastically different from that of radio signal, and thus acoustic channel models have to be incorporated; thirdly, underwater sensor networks are usually deployed in a three-dimensional space, while these simulators usually only support two-dimensional deployment. Thus, the unique characteristics of underwater sensor networks make the existing network simulators unsuitable.

We have developed a simulator, called Aqua-Sim, for underwater sensor networks. We choose NS-2 as the development platform since NS-2 is a very powerful, widely used, and open-source simulator. NS-2 provides efficient and convenient methods to configure network and nodes. Two languages are used in NS-2, C++ and Otcl. Users can use Otcl scripts to easily tune the parameters of protocols and algorithms implemented in C++. NS-2 also enables us to take the advantage of the abundant existing source codes. As NS-2 is widely used by the research community, it has rich wireless protocols and utilities that are potentially useful for underwater sensor networks. Most importantly, NS-2 is open-source, which allows public access and fast development.

Developed on the basis of NS-2, Aqua-Sim can effectively simulate acoustic signal attenuation and packet collisions in underwater sensor networks. Moreover, Aqua-Sim supports three-dimensional deployment. Further, Aqua-Sim can easily be integrated with the existing codes in NS-2. In this paper, we will present the design and implementation of Aqua-Sim and demonstrate the power of Aqua-Sim through several case studies.

II. Design and Implementation

In this section, we first give an overview of the Aqua-Sim design, discussing its motivation, structure, and class diagram. Then we present the implementation of Aqua-Sim, including physical layer models, MAC layer classes, and routing layer classes as well as some other commonly used classes.

A. Design Overview

In NS-2, the CMU wireless package was developed for terrestrial wireless networks. As discussed earlier, such a simulation package can not easily cope with underwater network environments. Specifically, as underwater acoustic channels feature long propagation delays and high attenuation, attenuation models for terrestrial radio channels can not be applied anymore. In addition, long propagation delays make the collision behaviors quite different from those in radio networks, which will require a different way to simulate the collisions in underwater sensor networks. Furthermore, three-dimensional deployment is not supported in the CMU wireless package. Besides these fundamentals, some basic protocols, such as MAC and routing, tailored for underwater network environments, are desired for the evaluation of any advanced protocol via simulations.

Motivated by the above needs, instead of patching on the existing simulation package of wireless networks, we develop a new simulation package, named as Aqua-Sim, for underwater sensor networks. In NS-2, Aqua-Sim is in parallel with the CMU wireless simulation package. Fig. 1 illustrates

0-933957-38-1 ©2009 MTS

the relationship between Aqua-Sim, CMU wireless package, and NS-2 basics. Aqua-Sim is independent of the wireless simulation package and is not affected by any change in the wireless package. On the other hand, any change to Aqua-Sim is also confined to itself and does not have any impact on other packages in NS-2. In this way, Aqua-Sim can evolve independently.

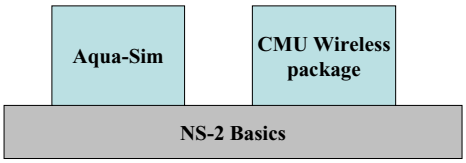


Fig. 1. Relationship between Aqua-Sim and other packages of NS-2

Aqua-Sim follows the object-oriented design style of NS-2, and all network entities are implemented as classes in C++. Currently, Aqua-Sim is organized into four folders, uw_common, uw_mac, $uw\_routing$ and uw_tcl. The codes simulating underwater sensor nodes and traffic are grouped in folder uw_common; the codes simulating acoustic channels and MAC protocols are organized in the folder of uw_mac. The folder $uw\_routing$ contains all routing protocols. The folder uw_tcl includes all Otcl script examples to validate Aqua-Sim. Users can use the examples as templates to devise new experiments.

In general, Aqua-Sim has three types of classes.

- *Network Entity Classes*: This type of classes represent concrete network entities. For example, a node's physical layer is represented as a "UnderwaterPhy" object and its broadcast MAC protocol is represented by a "Broadcast-MAC" object. These objects are required to fulfill their own functionalities and provide standard interfaces to the upper/lower layer network entities.
- *Pure Interface Classes*: This type of classes are purely virtual and can not be instantiated at all. However, they specify common interfaces and serve as the base classes for others. For example, the "UnderwaterMAC" class in Aqua-Sim provides a common interface to the MAC layer entities such as "BroadcastMAC" and "RMAC". Although the implementation of "BroadcastMAC" is quite different from that of "RMAC", they share the same interface to the physical layer and the logic link layer, which is specified by their base class "UnderwaterMAC".
- *Common Function Classes*: This type of classes provide some common functions to other classes and can be included into any classes in Auqa-Sim. Although these classes have no instantiations in the corresponding network node, they are quite important to Aqua-Sim. For example, "UW-hash-table" implements a highly efficient hash table which can be used by the routing and the MAC protocols.

Fig. 2 plots the class diagram of Aqua-Sim. In the figure, the "UnderwaterNode" object is the abstraction of the underwater sensor node. It encapsulates many useful information of the node such as its location and its moving speed. It is a global object and can be accessed by any object in Aqua-Sim. The "UnderwaterChannel" object represents the underwater acoustic channel. There is only one "UnderwaterChannel" object in the network and all packets are queued here before being delivered. The "UnderwaterChannel" object also provides public interface to upper layers and thus the object in the upper layer, such as a routing layer object, can easily get to know the channel properties.

All network entities are represented by their corresponding objects and interact with each other in the same way as specified in the protocol stack. For example, if the MAC protocol entity on node "A" wants to transmit a control packet to node "B", it will pass the packet directly to its "UnderwaterPhy" object which is the representative of its physical layer. This packet will then be passed to the unique "UnderwaterChannel" object, which will put the packet into a FIFO queue and calculate the propagation delay and attenuation. This packet then is passed to the "UnderwaterPhy" object on node "B". Finally, it will reach the MAC layer object on node "B".

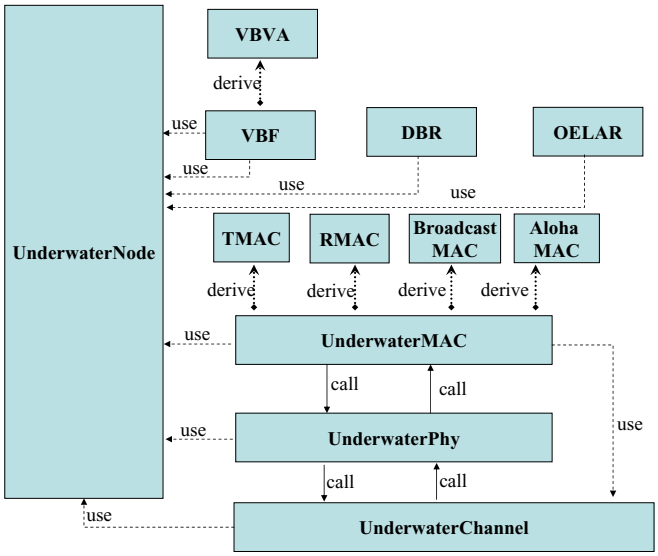


Fig. 2. Class diagram of Aqua-Sim

B. Physical Layer Models

1) Attenuation Model: Presently, Aqua-Sim adopts the signal attenuation model as in [3],

$$A(l,f) = l^k \times (10^{\frac{\alpha(f)}{10}})^l \tag{1}$$

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. k is set to 1 for cylindrical, 1.5 for practical and 2 for spherical spreading; $\alpha(f)$ is the absorption coefficient, which can be empirically expressed by the Thorp's equation [3] as

$$\alpha(f) = \frac{0.11 \times f^2}{1+f^2} + \frac{44 \times f^2}{4100+f^2} + 2.75 \times 10^{-4} \times f^2 + 0.003 \tag{2}$$

where f is in kHz. The propagation speed of acoustic signal adopted in Aqua-Sim is 1500 m/s. When a node sends a packet, Aqua-Sim first finds all the nodes in the transmission

range of the sender in the 3D space, then computes the propagation time of the packet to reach these nodes. Aqua-Sim simulates the packet propagation by introducing a delivery delay equal to the propagation time of the packet. Aqua-Sim provides users two ways to determine the transmission range: one way is to set the transmission power and the threshold of received power level; and the other way is to set the transmission range directly.

It should be noted that in Aqua-Sim, an independent class "UnderwaterPropagation" is designed to capture the attenuation characteristics of the underwater acoustic channel. When the "UnderwaterChannel" object receives a packet, it will call the functions in the "UnderwaterPropagation" object to calculate the signal attenuation. Such a design makes Aqua-Sim open to different attenuation models. To import a new attenuation model into Aqua-Sim, all we need to do is to derive a new class from the "UnderwaterPropagation" class and implement the attenuation model in the new class. The rest part of Aqua-Sim will not be affected.

We are now implementing a new underwater acoustic channel model which also takes the fast-varying multi-path propagation effects into consideration [15], [12], [8], [16], [2]. This model is well accepted in the community and represents the channel as follows:

$$h(\tau,t)=\sum_{p=1}^{N_p} A_p\eta_p(\tau-\tau_p(t)) \qquad with \tau_p(t)=\tau_p-a_pt. \quad (3)$$

where the channel consists of a total of N_p distinct paths, A_p is the path attenuation, $\eta_p(\tau)$ is the path specific filter whose Fourier transform captures the frequency-dependent propagation effect, $\tau_p(t)$ is the time-varying delay and a_p is the delay variation rate (also termed as Doppler scaling factor). The validity of this multi-path channel model has also been verified by the SPACE08 experiment near Marthar's vineyard, MA, Oct., 2008 [2].

2) Collision Model: Aqua-Sim can simulate packet collision on every node. Because of the long propagation delay of the underwater acoustic channel, different nodes in the network will see different network conditions. Packets which collide at one node may not collide at another node. In addition, packets may arrive at different nodes in different order.

In order to simulate the collision behaviors in underwater sensor networks accurately, we implement a new class named "IncomingChannel" and every node will maintain a different instantiation of this class. The packets in the "UnderwaterChannel" will first be copied into the "IncomingChannel" object on every node. Then these packets in the "IncomingChannel" object will be reordered according to their arrival time and then be forwarded to the physical layer. And at the physical layer, the "UnderwaterPhy" object will first scan the packet queue in the "IncomingChannel" and judge the current collision status.

If the packets overlap at the receiver side in the time domain, their received power levels, derived from the attenuation model and specified in the corresponding "UnderwaterPropagation" object, are compared. If the difference between the received power levels is less than a pre-defined threshold, then these packets are marked "collided", and all packets will be dropped. Otherwise, the packet with the highest received power level is correctly received to simulate the packet capture effect in the underwater acoustic channel. Since the "IncomingChannel" object on every node maintains its own copies of the received packets, collisions only happen among these local copies and will not affect the packet copies on other nodes. Thus, the effects of collisions are confined to the local node and will not affect other nodes, which exactly models the collision behaviors in underwater sensor networks.

The "UnderwaterPhy" object also provides interfaces to power on/off the transmitter and set the power level for transmitting and receiving. It also computes energy consumption and updates the remaining power level of a node. Node failures and channel failures are implemented as well.

C. MAC Layer

All classes related to MAC layer are included in the folder of uw_mac. Currently, four MAC protocols: Broadcast MAC, Aloha, T_u-MAC and R-MAC are implemented in Aqua-Sim. All these MAC protocols are derived from the same abstract base class "UnderwaterMac". Public interfaces that are common to all MAC protocols are specified in "UnderwaterMac". For example, if an object want to send a packet to the MAC layer object, it just call the common "Recv" interface, which is specified in "UnderwaterMac", without any knowledge of the detailed MAC layer implementation. Such a design strategy decouples the MAC layer interface and its implementation. New MAC protocols can be easily imported into Aqua-Sim by implementing the interface specified in "UnderwaterMac".

In the following, we brief the four MAC protocols in Aqua-Sim.

- *Broadcast MAC Protocol*: This is the barebone MAC protocol in Aqua-Sim: When a node has packets to send, it first senses the channel. If the channel is free, it broadcasts the packets. Otherwise, it backs off. Packets will be dropped if the number of back-off times exceeds the limit. When the receiver receives a packet, it does not need to send an ACK back to the sender. This protocol is simple yet efficient in low traffic networks. In addition, this protocol can take full advantages of the broadcast nature of the underwater acoustic channel and are suitable for geo-routing protocols such as VBF [21].
- *Aloha Protocol*: This protocol is based on the Aloha idea while tailored to underwater network environments: when a node has packets to send, it will send it directly without sensing the channel. The sender then will start a timer and wait for the response from the receiver. If the receiver receives the packet correctly, it will send an ACK back to the sender. If the sender receive an ACK before it times out, the sender knows that this packet has been successfully transmitted and starts to send the next packet. Otherwise, the sender will back off for some time and resend the same packet again.
- T_u*-MAC Protocol*: This is a modified T-MAC [17] protocol for underwater networks, refer to as T_u-MAC. We make three major revisions as follows: First, we

modify the active time to incorporate propagation delay, which is non-negligible in underwater sensor networks. Second, we modify the RTS/CTS method adopted in the original T-MAC since it is not suitable for the long delay underwater networks. In T_u-MAC, when a sender receives a CTS from the intended receiver, it waits until the CTS propagates through the whole transmission range of the receiver. Third, since carrier sensing does not make much sense in underwater sensor networks, T_u-MAC does not adopt this technique.

- *R-MAC Protocol*: R-MAC is proposed in [20] and is a reservation based MAC protocol. R-MAC is divided into three phases. In the initial phase, nodes measure the distances to neighbors by sending some small control packets. Based on the measurements, every node will randomly choose a period for its data transmission and inform others in Phase 2. In Phase 3, nodes cooperate with each other to schedule data transmissions to avoid collisions. Although R-MAC is much more complicated than other MAC protocols, it has the same interface as others and no changes are needed in other parts of Aqua-Sim.

D. Routing Layer

All classes related to the routing layer are included in the folder of $uw\_routing$. The implementation of all routing protocols follows the standard structure of existing routing protocols in NS-2. The parameters for the protocols can be adjusted through Tcl script. In order to support the advanced features of the routing protocols in underwater sensor networks, Aqua-Sim provides standard interfaces at almost every network layer. For example, geo-routing protocols based on node positions can easily get the location information from the interface of the "UnderwaterNode" object. Thus, Aqua-Sim provides a good platform to develop advanced routing protocols.

Currently, three routing protocols including: Vector based routing protocol (VBF) [21], Depth-base Routing (DBR) [22] and Q-learning-based Routing (QELAR) [7] are implemented in Aqua-Sim. We brief these protocols as follows:

- *VBF Protocol*: Vector-Based Forwarding (VBF) protocol [21] is a geographic routing protocol. Each node in the network is assumed to know its position. In VBF, the forwarding path follows a vector from the source to the target, which is called forwarding vector. The position information of the source, target, and forwarder is carried in the header of the data packet. When a node receives a packet, it calculates its distance to the forwarding vector. If the distance is less than a pre-defined threshold, called radius, this node is qualified to forward the packet. In VBF, the forwarding path is virtually a pipe from the source to the target, called forwarding pipe. VBF is very robust against mobile networks, error-prone channels and vulnerable sensor nodes.
- *Depth-Based Routing (DBR)*: DBR is a routing protocol based on a greedy forwarding algorithm [22]. It utilizes the depth information of underwater sensor nodes to forward packets from a source node to sinks deployed at water surface. Packets follow the rule of reducing the depth of forwarding nodes at each routing step toward the water surface. In DBR, once a packet is received, a node first gets the depth information of the packet's previous hop node which has been recorded in the packet. The receiving node then compares its own depth with the depth of previous hop node. If the receiving node's own depth is smaller than previous hop, namely the receiving node is closer to the water surface, it will be considered as a qualified candidate to forward the packet. Otherwise, it just simply discards the packet.
- *QELAR Protocol*: QELAR is proposed in [7]. It is an adaptive, energy-efficient, and lifetime-aware routing protocol based on reinforcement learning. By extracting needed information from underwater environment at runtime, learning agents are able to make optimal routing decisions to reduce the hop counts to the sink as well as to make the residual energy more evenly distributed, and therefore the lifetime of the whole network is largely prolonged. At the same time, the protocol can be easily tuned to balance the energy consumption and the residual energy distribution to make itself more flexible.

E. Other Classes

In the folder of uw_common, we define some common classes such as "UnderwaterNode" and "UnderwaterSink".

Class "UnderwaterNode" is derived from the base class "MobileNode" which represents the general mobile sensor node. "UnderwaterNode" defines the compositions of the underwater sensor node. For example, through the interface of "UnderwaterNode", users can specify the number and the types of the physical interfaces on a node. And users can also specify the routing and MAC protocol on this node. And this "UnderwaterNode" object keep references to all these components.

It should be noted that "UnderwaterNode" can specify its components to be classes outside of Aqua-Sim. For example, some surface buoys are equipped with two sets of transceivers. One is the acoustic transceiver which can communicate with the underwater nodes through acoustic channels. The other is the wireless transceiver which can communicate with the surface control center through the radio channels. Through the public interface of "UnderwaterNode" class, users can specify two types of physical interfaces: one is the underwater physical interface in Aqua-Sim and the other is the radio wireless interface in the original wireless simulation package, on the same node in Aqua-Sim. "UnderwaterNode" also addresses new features for underwater sensor nodes. For example, it allows underwater sensor nodes to be randomly deployed in a 3-D space and can move freely in three-dimensions.

In Aqua-Sim, all "UnderwaterNode" objects are put into a global list and are sorted according to their locations. This node list is managed by a global "GOD" object which is provided by NS-2.

Class "UnderwaterSink" is used to generate different traffic patterns. It also collects statistic information, such as packet delivery ratio, end-to-end delay, and energy consumption, etc., for underwater sensor nodes. Currently constant bit rate (CBR)

and exponential distribution traffic patterns are supported in "UnderwaterSink".

III. Case Studies

In this section, we present several case studies. By comparing simulation results with testbed results, we first show that Aqua-Sim can reproduce the real world with high fidelity. Then we demonstrate the flexibility of Aqua-Sim via some comparison studies of MAC protocols and routing protocols.

A. Fidelity Testing

In order to test the fidelity of Aqua-Sim, we design experiments to compare simulation results with real testbed results. The real tests are performed in Aqua-Lab [11], a lab testbed developed by the Underwater Sensor Network Lab at the University of Connecticut.

Fig. 3 shows the network topology for both simulations and real tests. Node 6 is configured as the receiver and all other nodes are sending data to node 6. This gives us a one hop, multi-source and single-sink topology. Micro-modem [5] used in Aqua-Lab provides a data rate at 80bps and we also set the data rate to 80bps in the simulations.

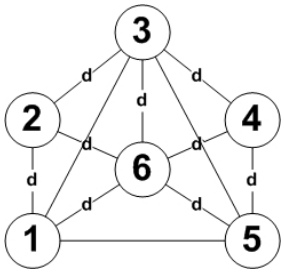


Fig. 3. Network topology for fidelity testing

We implement the Aloha protocol as described in Section II-C for both Aqua-Lab and Aqua-Sim. The packet length and the ACK message are set to 32 bytes. The maximal number of retransmissions is 3 and the input traffic of every node follows an exponential distribution. Two experiments are conducted to compare the simulations and real tests. In the first experiment, we fix the average input traffic of every node to be 0.02 packet per second and change the number of data senders from 1 to 5. In the second experiment, we fix the total average input traffic to be 0.1 packet per second and change the number of data senders n. Thus, the input traffic of every node is $\frac{0.1}{n}$.

The results are plotted in Fig. 4 and Fig. 5. From these figures, we can clearly tell Aqua-Sim can closely approximate the results of the real tests under different input traffic patterns. In other words, we can claim that Aqua-Sim could reproduce the real world with high fidelity.

Because of the relatively high cost of underwater sensor nodes, it is often cost forbidden to develop a testbed or deploy a real network for every research effort. As an alternative, Aqua-Sim provides us a powerful tool to conduct research in underwater sensor networks.

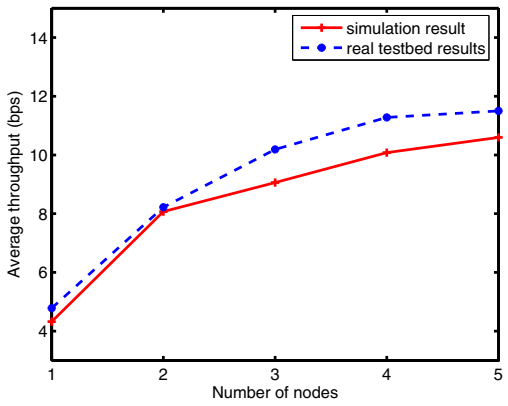


Fig. 4. Throughput with fixe input traffic per node

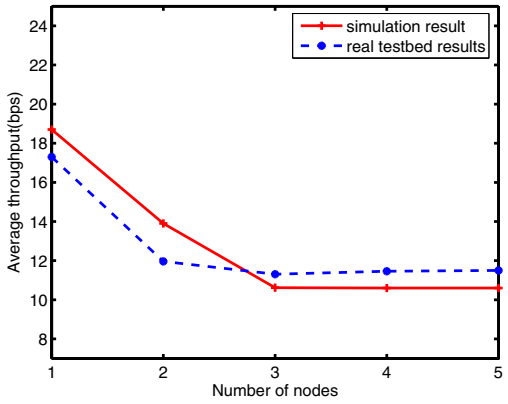
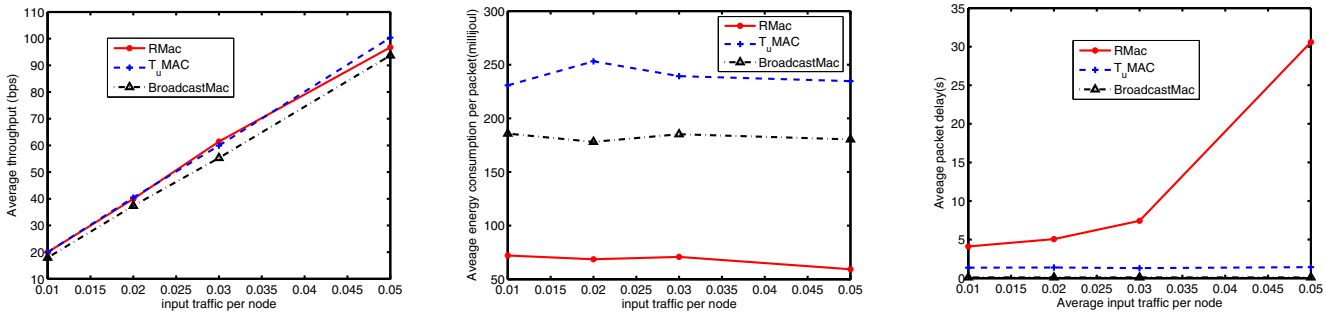


Fig. 5. Throughput with fixed total input traffic

B. Comparing MAC Protocols

In this set of simulations, we use the one-hop network as shown in Fig. 3 to compare three MAC protocols: R-MAC, T_u-MAC and Broadcast MAC. The packet length is set to 64 bytes and the data rate is set to 10kbps. We change the input traffic gradually from 0.01 to 0.05 packet per second.

We measure three metrics: average throughput, average energy consumption per packet, and average packet delay. Fig. 6 plots the corresponding results, from which we can observe that although the three protocols can achieve similar system throughput, R-MAC is much more energy efficient than the other two. This is because R-MAC measures the propagation delay and accurately schedules the node's transmission to completely avoid collisions. For T_u-MAC, an additional RTS/CTS handshaking process is involved for every data transmission, which makes its energy efficiency worst. However, the energy benefits of R-MAC do not come without any costs. The delay measurement and the scheduling process make its packet delay much larger than the other two. Thus, R-MAC is more suitable for networks with high demand on system energy efficiency and without strict packet delay requirement, while T_u-MAC and Broadcast MAC fit better in applications with low delay requirements.



(a) Average throughput (b) Average energy consumption per packet (c) Average packet delay

Fig. 6. Performance of MAC protocols

C. Comparing Routing Protocols

1) DBR vs VBF: In this set of simulations, we compare DBR with VBF. In the simulations, (underwater) sensor nodes are randomly deployed in a $500m \times 500m \times 500m$ 3-D space. There is one sink at the surface. While we assume that the sink is stationary once deployed, the sensor nodes follow the random-walk mobility pattern. Each sensor node randomly selects a direction and moves to the new position with a random speed between the minimal speed and maximal speed, which are 1 m/s and 5 m/s respectively. The data generating rate at the source node is one packet every 10 seconds, with a packet size of 46 bytes. The communication parameters are set as follows: the bit rate is $10k$bps; the maximal transmission range is 100 meters (in all directions); and the power consumption in sending, receiving, and idling mode are $2w$, $0.1w$, and $10mw$, respectively. The Broadcast MAC protocol is used in the MAC layer.

Fig. 7 plots the comparison results with respect to three metrics: packet delivery ratio, energy consumption, and average end-to-end delay. The results show that in sparse networks DBR has higher packet delivery ratio than VBF, while achieving similar energy efficiency (in dense networks, DBR is not an obvious winner). The cost paid by DBR is higher end-to-end delay. It should be noted that the results presented here are for one sink networks. It is expected that DBR has better performance in multiple-sink settings.

2) QELAR vs VBF: In this set of simulations, we compare QELAR with VBF. The maximum node velocity is set to 1m/s, data rate is set to 10kbps, and the data generation rate is set to 0.5 packets per second. Fig. 8 compares the delivery ratios of QELAR and VBF. As we can see that QELAR achieves much higher delivery ratio (near 100%) in that QELAR has its own implicitly acknowledging and retransmission schemes, and backtracking will be provoked when a node has no neighbor to forward the packet to. For VBF, however, the delivery ratio is lower than QELAR at all node densities especially when the network is sparse.

Fig. 9 depicts the energy consumption of QELAR and VBF. We can see that in a relatively sparse network, QELAR consumes more energy. This is because QELAR needs more retransmissions when the network is sparse, which consumes more energy. As the network becomes denser, less retransmission are required, and the energy consumption increases only because of the increase of total idle power. In VBF, because more and more nodes are involved as the number of nodes increases, the total energy consumption increases much faster than QELAR.

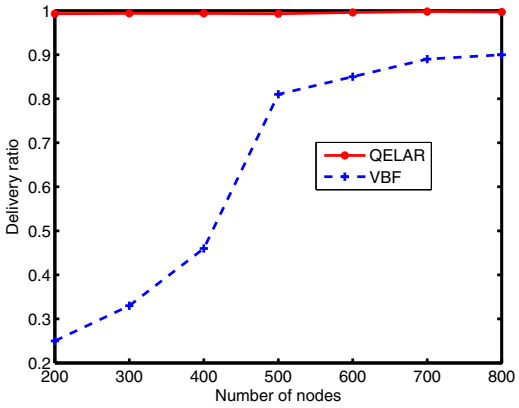


Fig. 8. Delivery ratio of QELAR and VBF

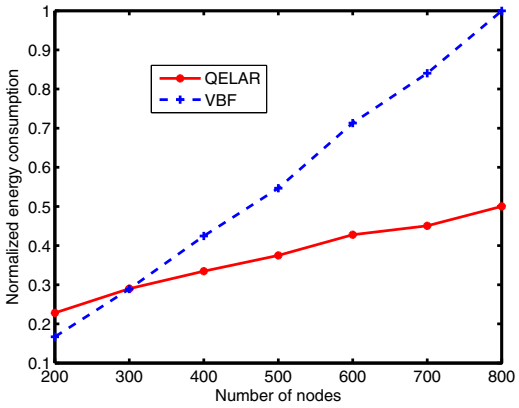
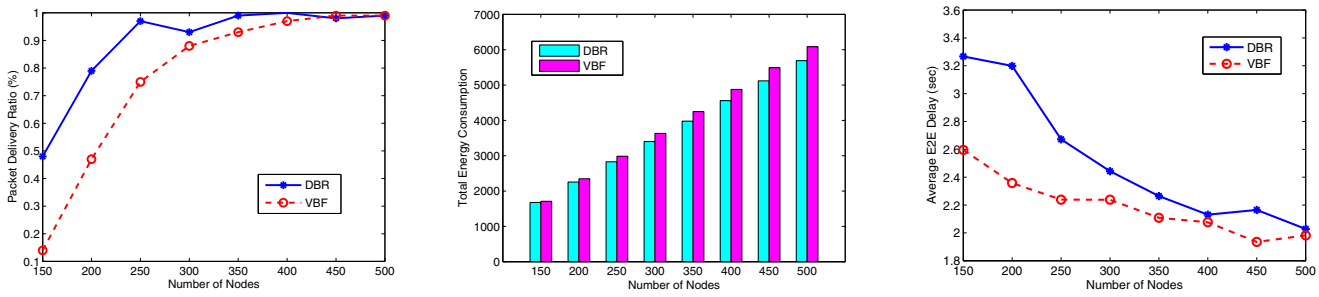


Fig. 9. Total energy consumption of QELAR and VBF

IV. Conclusions and Future Work

Due to the significant distinctions between underwater sensor networks and terrestrial radio networks, a new simulator for underwater acoustic networks is urgently needed. In this



(a) Packet deliver ratio (b) Energy consumption (c) Average end-to-end (E2E) delay

Fig. 7. Performance comparison of DBR and VBF

paper, we have presented the design and implementation of Aqua-Sim, a packet level underwater sensor network simulator based on NS-2. Through several case studies, we show that Aqua-Sim is a powerful simulation tool, with high fidelity and flexibility, for underwater networking research.

A Beta version of Aqua-Sim has been released in the Internet (http://uwsn.engr.uconn.edu/aquasim.tar.gz). To maintain and evolve a good network simulator, significant efforts will be required along the way. As some immediate future work, we plan to evaluate Aqua-Sim with more real experiment data, especially after incorporating advanced channel models.

REFERENCES

[1] I. F. Akyildiz, D. Pompili, and T. Melodia. Challenges for Efficient Communication in Underwater Acoustic Sensor Networks. *ACM SIGBED Review*, Vol. 1(1), July 2004.

[2] C. R. Berger, S. Zhou, J. Preisig, and P. Willett. Sparse channel estimation for multicarrier underwater acoustic communication: From subspace methods to compressed sensing. In *Proceedings of MTS/IEEE OCEANS Conference*, May 2009.

[3] L. Berkhovskikh and Y. Lysanov. *Fundamentals of Ocean Acoustics*. New York, Springer, 1982.

[4] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou. Challenges: Building Scalable Mobile Underwater Wireless Sensor Networks for Aquatic Applications. *Special Issue of IEEE Network on Wireless Sensor Networking*, May 2006.

[5] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball. The whoi micro-modem: An acoustic communications and navigation system for multiple platforms. In *Proceedings of IEEE Oceans*, 2003.

[6] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li. Research Challenges and Applications for Underwater Sensor Networking. In *IEEE Wireless Communications and Networking Conference*, Las Vegas, Nevada, USA, April 2006.

[7] T. Hu and Y. Fei. QELAR: a q-learning-based energy-efficient and lifetime-aware routing protocol for underwater sensor networks. In *27th IEEE Inter. Performance Comput. and Communi. Conference*, Austin, Texas, USA, December 2008.

[8] B. Li, S. Zhou, M. Stojanovic, L. Freitag, and P. Willett. Multicarrier communication over underwater acoustic channels with nonuniform doppler shifts. *IEEE Journal on Ocean Engineering*, 33(2), Apr 2008.

[9] NS-2. http://www.isi.edu/nsnam/ns/ .

[10] OPNET. http://www.opnet.com.

[11] Z. Peng, J.-H. Cui, B. Wang, K. Ball, and L. Freitag. An underwater sensor network testbed: Design, implementation and measurement. In *In Proceedings of Second ACM International Workshop on UnderWater Networks (WUWNet'07)*, 2007.

[12] J. Preisig. Acoustic propagation considerations for underwater acoustic communications network development. *ACM SIGMOBILE Mobile Comp. Commun. Rev*, 11(4), Oct 2007.

[13] J. Proakis, E.M.Sozer, J. A. Rice, and M.Stojanovic. Shallow Water Acoustic Networks. *IEEE Communications Magazines*, pages 114–119, November 2001.

[14] E. M. Sozer, M. Stojanovic, and J. G. Proakis. Design and Simulation of an Underwater Acoustic Local Area Network. *Northeastern University, Communications and Digital Signal Processing Center, Boston, Massachusetts,*, 1999.

[15] M. Stojanovic. On the relationship between capacity and distance in an underwater acoustic channel. *ACM SIGMOBILE Mobile Comp. Commun. Rev*, 11(4), Oct 2007.

[16] M. Stojanovic and J. Presig. Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Communications Magazine*, Jan 2009.

[17] T. van Dam and K. Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *ACM SenSys'03*, Los Angeles, California, USA, November 2003.

[18] G. Xie, J. Gibson, and L. Diaz-Gonzalez. Incorporating Realistic Acoustic Propagation Models in Simulation of Underwater Acoustic Networks: A Statistical Approach. In *MTS/IEEE Oceans'06*, Boston,MA, September 2006.

[19] G. G. Xie and J. Gibson. A Networking Protocol for Underwater Acoustic Networks. In *Technical Report TR-CS-00-02*, Department of Computer Science, Naval Postgraduate School, December 2000.

[20] P. Xie and J.-H. Cui. R-MAC: An Energy-Efficient MAC Protocol for Underwater Sensor Networks. In *Proceedings of International Conference on Wireless Algorithms, Systems, and Applications (WASA)*, Aug. 2007.

[21] P. Xie, J.-H. Cui, and L. Lao. VBF: Vector-Based Forwarding Protocol for Underwater Sensor Networks. In *Proceedings of IFIP Networking'06*, Coimbra, Portugal, May 2006.

[22] H. Yan, Z. Shi, and J.-H. Cui. DBR: Depth-Based Routing for Underwater Sensor Networks. In *Networking 2008*, 2008.