

# Improving TCP Performance over Underwater Sensor Networks

Naoya IIKUBO<sup>†</sup> Masayuki NAKATSUKA<sup>†</sup> Jiro KATTO<sup>†</sup> Hayato KONDO<sup>‡</sup>  
<sup>†</sup>Graduate School of Science and Engineering, Waseda University

<sup>‡</sup>Graduate School of Marine Science and Technology, Tokyo University of Marine and Technology  
 E-mail: <sup>†</sup>{iikubo, nakatsuka, katto}@katto.comm.waseda.ac.jp <sup>‡</sup>hkondo@kaiyodai.ac.jp

## 1. Introduction

Recently, Underwater Sensor Network (USN) has attracted attention as an observation technology of the ocean. However, the data transmission in this underwater environment suffers from various influences such as bandwidth usage limitation, surrounding noise, and large acoustic propagation delays. Therefore, communication itself is a difficult problem. Fig. 1 shows the example of composing USN. In Fig. 1, the observational data are acquired by underwater sensors (uw-sensors) in the bottom of the sea acquired is done and the data transmission is done by a multi-hop manner by using the acoustic links for an underwater sink (uw-sink)[1].

The TCP protocol of a conventional window-based mechanism can't communicate efficiently because of underwater characteristics. Because bit error rates are high and propagation delays are large, it takes long time to raise the window size and the time-out happen. As a result, it leads to decrease of TCP throughput.

Therefore in this paper, we examine performance improvement methods of the TCP protocol in the underwater multi-hop environment of the TCP. First, we consider that TCP-Hybla[2], which assumes its use on the satellite link of which RTT is large, is also effective in the underwater environment. We evaluate its performance of the multi-hop communication in the underwater environment. Second, [3] shows that, in the wireless multi-hop communication, it is effective in throughput improvement to limit the maximum congestion window size. Therefore, we try the maximum window size control in the multi-hop communication of the underwater environment.

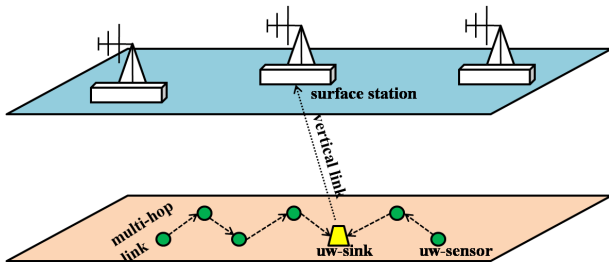


Fig. 1: Example Architecture of USN

## 2. Backgrounds

### A. TCP-Hybla[2]

Hybla is a congestion control method that is effective when propagation delays are large. The window size is controlled by the following parameters.  $W$  is the window size,  $RTT$  is round trip time, and  $\rho$  is  $RTT/RTT_0$ , of which default value is given by 0.025[sec].  $SS$  shows a slow start phase and  $CA$  shows a congestion avoidance phase.

$$W_{i+1} = \begin{cases} W_i + 2^\rho - 1 & SS \\ W_i + \rho^2 / W_i & CA \end{cases} \quad (1)$$

### B. Limitation of Maximum Window Size

When a window size becomes larger for TCP in multi-hop communication, contention increases in the MAC layer and packet loss rate due to buffer overflow decreases. As a result, throughput decreases. To alleviate this problem, [3] proposes a method which limits a maximum window size. It leads to avoiding contention in the MAC layer and increases the throughput.

## 3. Experiments

We simulate above methods by using an underwater model for ns-2 [4] offered by UIUC [5]. The simulation assumes that the depth of uw-sensors is fixed to 1000m at the bottom of the sea in Fig. 1.

Fig. 2 shows the simulation topology. Distances between each node are equally 70m. The transmission node is node 0, and the reception node is node  $N$ . The reference TCP throughput over multi-hop underwater links is measured by using NewReno for transport protocol and the link bandwidth is 150 Kbps. Effects of the TCP-Hybla and the maximum window size control are then evaluated.

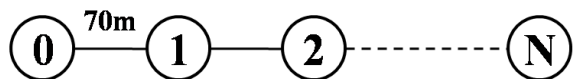


Fig. 2: Simulation topology

### A. TCP-Hybla Performance over USN

TCP throughput is measured when  $N$  is changed from 1 to 6. The packet size is assumed to be 512bytes and the throughputs of NewReno and Hybla are compared.

Parameter  $RTT_0$  of Hybla is assumed to be 0.7[sec]. Fig. 3 shows the simulation result.

From the simulation result, when the number of hops is from 2 to 5, the throughput of Hybla is higher than NewReno. This is because  $RTT$  increases as the number of hops increases, the  $\rho$  of Eq.(1) grows, and the window size of Hybla can be increased enough. However, in one hop case, the window size of Hybla is not larger because  $RTT$  is small, and the throughputs of NewReno and Hybla are almost the same.  $RTT$  is about 710[ms] by 70m. However,  $RTT$  changes largely and goes up to about 10[sec] due to the retransmission and the time-out. On the other hand, packet arrival rate of Hybla decreases as the number of hops increases (i.e.  $RTT$  increases) although its throughput is still higher than that of NewReno. This is because too large window size of Hybla causes heavy buffer overflow.

Because the window control of Hybla is influenced by constant parameter  $RTT_0$ , it is necessary to choose suitable  $RTT_0$  according to the network. The buffer overflow can be reduced by selecting suitable  $RTT_0$  but adequate setting of  $RTT_0$  is still a difficult problem because  $RTT$  is not stable in the USN environment.

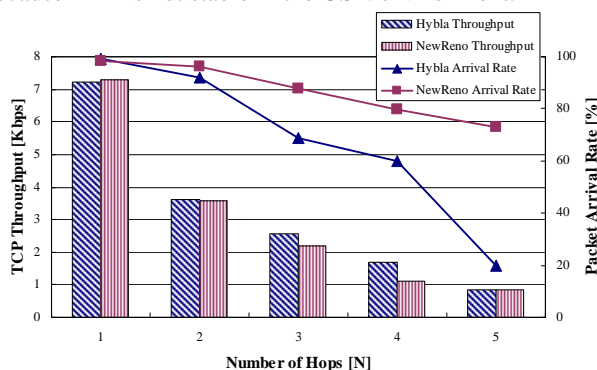


Fig. 3: TCP-NewReno vs. TCP-Hybla

## B. Effect of Maximum Window Control

In this experiment, the number of hops is fixed to 4, and three packet sizes are tried from 256bytes to 1024bytes. We then limit the maximum window size from 1 packet to 10 packets, and measure the throughput of NewReno. Fig. 4 shows its simulation result.

Throughputs are improved in all the packet size cases. We compared them with the case when the maximum window size is not limited shown in Fig.3, and found that it is possible to improve the throughput by 7% at 256bytes, 54% at 512bytes and 35% at 1024bytes.

It is also observed that, as a maximum window size increases, the throughput and the packet arrival rate decrease and reach a constant value. This is a similar result to that of [3]. The reason is that the window size does not increase above the value. When the number of transmission packets increases, packet collision comes to

happen in multi-hop links. However, when the window size is limited to small, efficient communication is not possible. From this simulation, we understood that the limitation of the maximum window size decrease possibilities of the congestion and the packet collision, and lead to improvement of the throughput. Though we omitted, window limitation results of Hybla had not shown significant improvement yet.

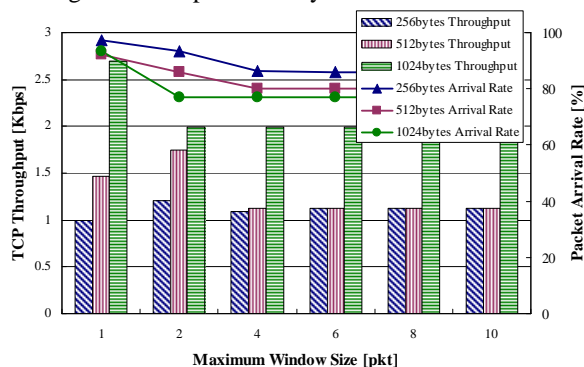


Fig. 4: TCP throughput for different maximum window size

## 4. Conclusions

In this paper, we evaluated the TCP throughput in the multi-hop underwater communication. TCP-Hybla and the maximum window size control are evaluated, and their effectiveness is validated by the simulations result. However, TCP-Hybla has a problem of decrease in packet arrival rate due buffer over flow and needs to set suitable parameter  $RTT_0$ . The maximum window size control for NewReno needs adequate choice of the maximum window size according to packet size, but it shows stable improvement in throughputs and packet arrival rates as long as the maximum window size is kept small in the USN environment.

## References

- [1] I. F. Akyildiz, D. Pompili, T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges", *Ad Hoc Network*, Vol1.3, pp.257-279, Feb. 2005.
- [2] C. Caini, R. Firrincieli, "TCP Hybla: A TCP Enhancement for Heterogeneous Networks", *Int. J. Satell. Comm. Network*, Vol1.22, pp.547-566, Sep.2004.
- [3] Z. FU, P. Petros, S. Lu, L. Zhang, M. Gerla, "The Impact of Multihop Wireless Channel on TCP Throughput and Loss", *IEEE Trans. Mobile Computing*, Vol1.4, pp.209-221, 2005.
- [4] "The network simulator -ns2-" <http://www.isi.edu/nsnam/ns/>
- [5] A. F. Harris III, M. Zorzi, "Modeling the Underwater Acoustic Channel in ns2", *NSTools 2007*, Oct. 2007.