Short Paper: Performance Analysis of a MACA based Protocol for Adhoc Underwater Networks

Shiraz Shahabudeen
ARL, National University of Singapore
shiraz@nus.edu.sg

Mehul Motani
ECE, National University of Singapore
motani@nus.edu.sg

ABSTRACT
This paper presents novel performance analysis of a MACA-based MAC protocol for underwater networks. MACA is very popular in terrestrial radio networks (such as 802.11) and many papers have presented its performance analysis. However, previously published analysis of such protocols for radio wireless networks typically do not account for propagation delay, high detection and decoding errors that characterize underwater networks and also do not derive closed form expressions for essential metrics. Some analysis on similar protocols has been done for underwater networks as well but there are significant differences in protocol models and analysis methodology. The analysis in this paper incorporates these additional factors and we derive analytical closed form expressions for mean service time and throughput for such a protocol. The analytical results match simulation results well. We also present preliminary sea trial results of modem implementation of the protocol that match the analysis well.

Keywords
Medium Access Control, Adhoc Underwater Networks, Performance Analysis, MACA, FAMA

1. INTRODUCTION
THIS paper presents analysis of a MACA based protocol for use in underwater acoustic networks (UAN). For the single channel distributed network scenario, one of the best dynamic and adhoc protocol family is MACA or related protocols such as FAMA [2, 3], DACAP [4] etc. In the protocol used here, an RTS/CTS handshake is followed by *B* DATA packets and an ACK packet that indicates the packets successfully received. There are retries until a packet is successfully transferred. MACA with packet trains that employ ACKs after every packet is not efficient for underwater applications due to round trip delay. The objective is a saturated load analysis as done in [1] which can then be used in queuing theory based analysis.

We review some of the most important prior related work and highlight some of the key differences. In [2], the FAMA protocol, a close variant of MACA is analyzed. Packet collisions are the only source of error, and bit errors are not considered. A three-way handshake with no ACK model is used and saturated load analysis is not presented. In [1], IEEE 802.11 DCF (variant of MACA) is analyzed. However, packet detection losses and BER losses were not considered, the back-off algorithm used is different (freezing back-off) and the service time equation derived is not in closed form. In [3], a UAN oriented analysis of FAMA is presented. A key difference with the protocol analyzed here is that the data packet train (or batch) concept is different (each DATA packet is acknowledged one at a time). The expression for throughput does not show the impact of batch size and the relationship to back-off window size is not captured. Also the throughput analysis is not a saturated load analysis as done here. ACK based DATA re-transmissions for reliability was not factored into the analysis in the reviewed papers. The main contribution of this paper is a reasonably accurate analytical service time model for a MACA-based protocol, whose results match simulations and acoustic modem field trials well. The protocol includes a novel ARQ variation which we term Early-ACK.

WUWNet'09, November 3, 2009, Berkeley, CA, USA.

2. ANALYSIS PRELIMINARIES
In this section we outline the important system aspects and the performance measures used.

2.1 Key System Aspects
We use a packet model with a fixed length detection preamble at the start of each packet. The detection probability P_d is dependent on the nature of the preamble. We define the packet decoding probability as P. In this paper, for simplicity, we use fixed packet time length L and P for control and data packets (multiple data packets can be sent as a batch), and assume static channel conditions and fixed FEC. Other parameters include number of nodes N in the collision domain (we assume no hidden nodes [2]), propagation delays D (consider maximum delays), RTS back-off window size W, and uniform back-off. Timers are used to wait for CTS and ACK (t_A). These timers are related to D and to control packet length L to give enough time for the round trip delay as

$$t_A = 2D + 2L \tag{1}$$

In a single collision domain scenario, the number of nodes N is the total number of nodes. In a multiple collision domain (multi-hop) scenario, N is the number of neighbors each node effectively contends with. The time index is denoted as t. The probability that a packet is detected and decoded correctly k, is

$$k = P_d P \tag{2}$$

2.2 Performance Measures

We define mean batch service time s_b as the average delay from the time a batch is intended for transmission (RTS contention starts) until it is considered successfully transmitted, i.e. until the first ACK is received for the batch. Another important performance metric is saturation throughput – the throughput of the network when the queue is saturated or always has data to transmit [1]. Such a measure is valid for file transfer applications and is also a measure of efficiency or channel utilization. We define normalized throughput T as the number of packets successfully transferred per unit time normalized by the system capacity, which is 1 packet in time L. B packets are sent as a batch in time s_b by definition, and of these, only k succeed due to BER and detection losses. Therefore the normalized throughput T per node is

$$T = \frac{kB/s_b}{(1/L)} \tag{3}$$

3. EXPECTED SERVICE TIME AND THROUGHPUT

Here we derive expressions for the expected batch service time s_b and throughput T. Time slotting is not necessary for the MACA based protocol used here as they function well without it. Our simulations (results not shown here for brevity) showed that in the scenarios considered in this paper, slotting makes very little difference to performance. However, we use the slotted model here, similar to the FAMA model in [3], primarily for its analysis simplicity. Similar to the definition in [3], during the RTS contention phase, the slot duration l is defined as

$$l = L + D \tag{4}$$

This allows collisions to be contained within slot boundaries. For $D \leq L$, packets transmitted in the same slot will at least partially collide. For example, for the ARL modem, highly robust control packets have duration L=0.6s. Thus the model is very effective for D of the order of 0.6s (up to 900meters range). For D >> L, packets in the same slot might not collide and the analysis is expected to give a conservative bound. For D >> L, time slotting (as defined above) as a protocol feature might also prove to be ineffective, and the un-slotted version could potentially outperform the slotted version.

In the RTS contention algorithm used here, a node starts with a uniformly selected back-off time slot in the integer range of [1, W]. The actual contention window time period is Wl. Also note that t_A as defined in (1) is $2l$. When the back off timer expires, a RTS is sent. Once an RTS is sent, the CTS timer t_A starts. If the timer expires before reception of CTS, the RTS back-off procedure starts again. Once CTS is received, DATA train is sent followed by wait for ACK. If ACK is not received, the RTS cycle repeats. Reception of RTS/CTS packets and a possible DATA frame while waiting to send RTS triggers Virtual Carrier Sense (VCS). Successful DATA transmission for any one node restarts RTS contention cycle for all. Note that 802.11 uses freezing back-off which is described in [1] whereas we use a constant window. Also, this protocol does not use Physical Carrier Sense (PCS) whereas it is used in 802.11.

For simplicity of analysis, we assume that no collisions happen during the CTS period since VCS start due to RTS reception. So

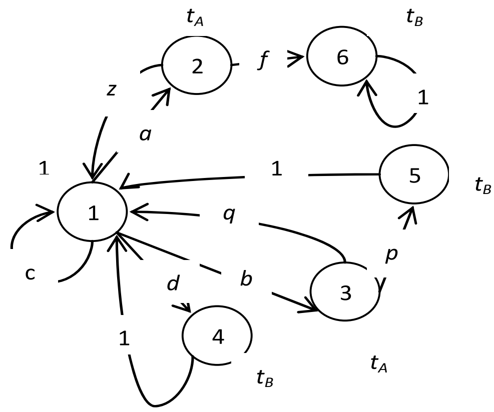

Figure 1. Markov chain model

in our analysis model, CTS loss will only be due to BER and packet detection probability. This simplification is quite valid due to the assumption of no hidden nodes. If the transmitter does not get CTS, it restarts contention window for RTS. Any other node which had received the RTS does a VCS for CTS. It resets and restarts contention if CTS does not arrive. Thus until one node gets a CTS and DATA transmission starts, this process will continue. In order to handle the case of some nodes missing the winning CTS and interfering with the DATA phase, all nodes monitor for DATA packets and DATA packets contain information of how many packets remain in the batch. This helps nodes that missed RTS/CTS to regain VCS with a probability close to 1 after a few DATA packets are sent (probability of getting at least one packet after n packets are sent is 1- $(1\text{-}P_dP)^n$ which get close to 1 as n increases). Thus the contention cycle synchronization is maintained. This is similar to the NAV concept used in 802.11.

We can represent the protocol using the model in Figure 1. Circles with enclosed numbers are states. The expressions along the arrows show the transition probability. The expressions placed next to the state circles is the duration spend in that state. In analysis, state transitions will be represented as a pair such as (g,h) for a transition from state g to h. State transition probability will be represented as P(g,h).

We introduce some new variables in (5) to make the presentation of Figure 1 and the matrices in (7) compact.

$$\begin{aligned}
&y = (N-1)\frac{1}{W}\left(1-\frac{1}{W}\right)^{N-2} \\
&p = k^2; \quad q = 1-k^2; \quad f = k^2\left(1-\frac{1}{W}\right)^{N-1} \\
&z = 1-f; \quad a = 1/W; \quad b = \left(1-\frac{1}{W}\right)ky \\
&c = \left(1-ky(1+k-k^2)\right)\left(1-\frac{1}{W}\right) \\
&d = \left(1-\frac{1}{W}\right)(1-k)k^2 y
\end{aligned} \tag{5}$$

The start of RTS contention cycle is at state 1. At the start of a new slot, a node has a probability of sending a RTS, i.e. P(1,2) = $1/W$. Once a RTS is sent, node is in state 2, waiting for t_A time slots for CTS to arrive. If CTS arrives, it goes to state 6 and transmits a batch of DATA, and including round trip time for ACK, the total duration is

$$t_B = BL + t_A = BL + 2l \tag{6}$$

The probability that the RTS transmitted in a given slot encounters no collision from any other node is $(1\text{-}1/W)^{N\text{-}1}$, i.e. no other node transmits a RTS in the same slot. CTS will be successfully received if apart from having no collisions, RTS is received at the receiver (probability k) and the CTS in turn is received at the transmitter (probability k) with a combined probability of k^2. This is shown in Figure 1 as P(2,6). Else transition (2,1) happens as shown.

If a RTS is not sent (probability 1-1/W), the current node counts down the RTS timer by one slot. During this back-off period, the probability that one of the N-1 neighbors has a successful RTS transmission is (N-1) (1/W) $(1\text{-}1/W)^{N\text{-}2}$ using same arguments as in last paragraph. And with k being the RTS detection probability, the current node could receive a RTS from another node with probability k (N-1) (1/W) $(1\text{-}1/W)^{N\text{-}2}$. In this case transition (1,3) occurs as shown (Note the additional 1-1/W term for probability of not having sent RTS as stated earlier). In state 3, it awaits CTS for time t_A. Thereafter if CTS is successful (probability k^2, since both RTS needs to be independently received by recipient and CTS received by current node), it goes to state 5 for batch VCS, following which it goes back to state 1 with probability 1 immediately. CTS failure in state 3 with probability $1\text{-}k^2$ takes the system back to state 1.

If during back-off as described in last paragraph (probability 1-1/W), CTS is received directly, transition (1,4) occurs. As before, the probability that at least one of the N-1 neighbors has a successful RTS transmission is (N-1)(1/W) $(1\text{-}1/W)^{N\text{-}2}$. But this RTS was missed (probability 1-k) but CTS was received (RTS received at other node and CTS received by node under consideration with probability k^2). Thus P(1,4) is as shown. In state 4 VCS for batch reception for time t_B occurs and goes back to state 1 thereafter with probability 1. If system is backing off and either RTS or CTS from others is not received as stated above, it goes back to state 1 as shown in (1, 1). A Markov chain matrix M can be used to represent this as follows using P(a,b) as shown in Figure 1. Q is the transient state matrix.

$$M = \begin{bmatrix} c & a & b & d & 0 & 0 \\ z & 0 & 0 & 0 & 0 & f \\ q & 0 & 0 & 0 & p & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} Q = \begin{bmatrix} c & a & b & d & 0 \\ z & 0 & 0 & 0 & 0 \\ q & 0 & 0 & 0 & p \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{7}$$

The fundamental matrix F is then,

$$F = (I - Q)^{-1} \tag{8}$$

Let E(m, n) be the expected number of times the system in the state n after starting from state m. E(1,n) is the expected number times the state n will be visited if the chain starts in state 1. Using standard Markov Chain theory E(1,n) = $F_{1,n}$.

Now we look at the process of re-transmissions. After the process from state 1 to 6 in Figure 1, ACK is sent by receiver. Note that ACK will be sent if any one packet in the train gets through. For large number of packets in the batch, we can assume that the receiver will get at least one packet and hence

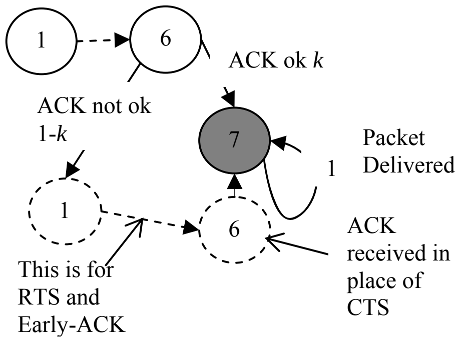

Figure 2. Markov chain for computing s_b

we assume it will send back an ACK with probability 1 at the end of the DATA round. Probability of getting at least one packet is $1\text{-}(1\text{-}k)^B$. For $P_d = P = 0.9$ and $B = 5$, P (get at least one packet) = 0.9998. Thus for most batch sizes we consider, we assume this to be 1 for simplicity, and get meaningful results.

A novel enhancement is to send ACK instead of CTS if RTS is repeated for the same packet train. This happens when ACK is lost. We call this the Early-ACK feature, as shown in Figure 2. With the addition of the Early-ACK feature, we need to modify the delay t_B spent in states 4 and 5 in Figure 1, since instead of CTS, some of it could be Early-ACKs. Since the probability of ACK not being received is 1-k, the percentage of time taken by early ACKs is (1-k)/(1+(1-k)) and this has no additional batch transmission time. Or, the percentage of batch transmission time is 1/(1+(1-k)). Let t_{B_VCS} be the modified VCS delay in states 4 and 5.

$$t_{B\_VCS} = t_B / (1 + (1 - k)) \tag{9}$$

Let the time until successful reception of CTS from state 1 to state 6 of Figure 1 be s_{CTS} (not counting the batch transmission time t_B in state 6). Then,

$$\begin{aligned} s_{CTS} &= (l)E(1,1) + t_A E(1,2) + t_A E(1,3) \\ &\quad + t_{B\_VCS} E(1,4) + t_{B\_VCS} E(1,5) \\ &= \left[\frac{l}{k^2 W} \left(\frac{W}{W-1} \right)^N \left(W^2 + W - 2 \right) + \frac{2(N-1)l}{k} \right] \\ &\quad + t_{B\_VCS}(N-1) \end{aligned} \tag{10}$$

As shown in Figure 2 using dotted circles, s_{CTS} is also the time taken to get an Early-ACK if the initial ACK fails after a batch transmission from a given node, since it uses the same process from state 1 to state 6 (excluding the batch transmission delay in state 6). Thus, based on Figure 2, the total batch service time s_b from state 1 to final state 7 (until Early-ACK, grayed circle) can now be computed using expected time from state 1 to 6 (including batch transmission) and additional time s_{CTS} for Early-ACK as follows,

$$\begin{aligned} s_b &= s_{CTS} + t_B + (1-k)s_{CTS} \\ &= (1 + (1-k))s_{CTS} + t_B \end{aligned} \tag{11}$$

3.1 Throughput for Reliable Transfer

Using equations (3) and (11), we can get the analytical total network throughput, i.e. NT. For L = 0.5, N =3, D = 0.5 and W = $N+1$ (heuristic), we show the throughput variation with batch

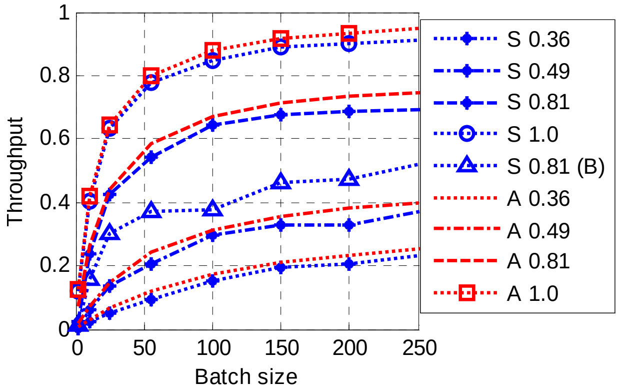

Figure 3. Network throughput vs. batch size for k = 0.36, 0.49, 0.81 and 1.0, simulations (S), analysis (A)

size B for different values for k in Figure 3. Simulation results are also shown in Figure 3. The simulator details can be found in [5]. In the simulation model, each node uses another fixed node as recipient. The nodes were randomly spread in a 500m by 500m area, thus restricting the maximum propagation delay to within 0.5s as used in the analytical computations.

The analytical results give close results to simulations. An example result without the Early-ACK feature marked as "S 0.81 (B)" for k=0.81 shows how Early-ACK feature improves performance (shown in "S 0.81"). For k = 1, i.e. no detection and decoding errors, the protocol is affected only by collisions and as shown, throughput can be made arbitrarily close to 1. The throughput has a saturation type behavior at higher batch sizes. Using (3) and (11) saturation throughput behavior is

$$T_{Lim\ B \to \infty} = \frac{k}{N} \tag{12}$$

If k=1, T converges asymptotically to 1/N.

4. MODEM TRIAL RESULTS

Here we present some acoustic modem sea trial results along with simulation and analytical results. Trial details can be found in [5]. An important aspect is that the same MAC protocol C code runs in the modem and the simulator through a unified simulator and modem software interface. Parameters used were maximum delay D of 0.4s to emulate trial distances, contention time window Wl =10s and batch sizes B=5, 10, 40. Saturated traffic model is used. In the sea trials, estimated P_d = 1 and P = 0.9. There is a crucial different aspect in the modem. It uses a high power amplifier (HPA) for transmission which takes about 300ms to settle after turn on, before transmission. We term this delay as t_{HPA}. When HPA is turned on, no receptions are possible. For batch transmission, HPA is turned on/off only at the start and end of the batch. This HPA behavior is captured into the simulations. Thus for the analysis, even though the actual packet duration (highly robust control or data packets in the ARL modem) is 0.6s, an effective time period of 0.9 seconds needs to be considered in the contention part, i.e. L=0.9. During batch transmission, since HPA is turned on/off only at the beginning and end, we define a different data packet length L_d = 0.6. Thus we modify (6), as follows

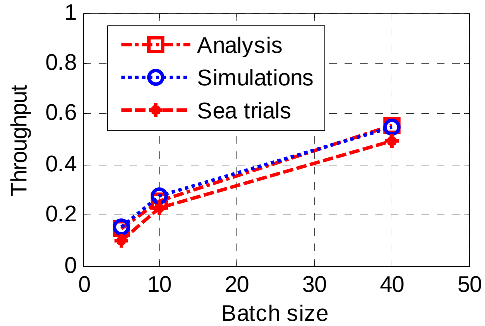

Figure 4. Analysis comparison with sea trials and simulations (also in [5])

$$t_B = BL_d + 2l + t_{HPA} \tag{13}$$

L_d is also used in place of L in (3). The results are shown in Figure 4. Here also, the analysis closely compares with simulations. It also presents a good match with the sea trial results. The sea trials give strong validation for both simulation and the analysis.

5. CONCLUSION

We proposed a protocol enhancement called the Early-ACK for a MACA based underwater MAC protocol for improving reliable delivery throughput. We derived a good analytical model for the protocol, that can help analyze the impact of any of the parameters N, D, L, B, k, W, t_A etc on expected service time and throughput and specifically illustrated the impact of parameters B (batch size) and k (detection and decoding probability) on throughput. We note that for very long latency regimes, the slotted version of the protocol could be less effective, and the analysis results less accurate. We implemented the protocols in acoustic modems, and medium range field trials corroborate the simulations and analysis quite well. The analytical model of the protocol can greatly help in understanding the protocol's dependence on environmental and system parameters without requiring simulations and will form the basis of a detailed queuing analysis of the protocol in future.

6. REFERENCES

[1] Bianchi, G. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, *18* (3). 535-547.

[2] Garcia-Luna-Aceves, J.J. and Fullmer, C.L., Performance of floor acquisition multiple access in ad-hoc networks. in *Third IEEE Symposium on Computers and Communications, ISCC '98*, (1998), 63-68.

[3] Molins, M. and Stojanovic, M., Slotted FAMA: a MAC protocol for underwater acoustic networks. in *MTS/IEEE OCEANS'06*, (2006).

[4] Peleato, B. and Stojanovic, M. A MAC protocol for Ad Hoc Underwater Acoustic Sensor Networks *WUWNet'06*, 2006.

[5] Shahabudeen, S., Chitre, M., Motani, M. and Low, A. Unified Simulation and Implementation Software Framework for Underwater MAC Protocol Development *MTS/IEEE Oceans '09*, Biloxi, US, 2009.