Analysis of a high performance MAC protocol for underwater acoustic networks

Shiraz Shahabudeen, *Member, IEEE,* Mehul Motani, *Member, IEEE,* and Mandar Chitre, *Senior Member, IEEE*

***Abstract*—**

Multiple Access with Collision Avoidance (MACA) is a popular Medium Access Control (MAC) protocol for terrestrial networks (e.g. 802.11). Underwater Acoustic Networks (UANs) differ from terrestrial networks as they are characterized by long propagation delays and higher data loss. Most of the published analysis for MACA in terrestrial networks ignores these factors, and therefore cannot be directly applied to UANs. As a result of the high data loss in UANs, it is common to implement reliability at a link level, rather than end-to-end acknowledgements. Keeping this in mind, we present a Markov chain analysis for a reliable variant of the MACA protocol for ad hoc UANs and derive closed-form expressions for mean service time and throughput. We show that the network performance is vastly improved with a few changes to the protocol, and propose a novel MACA-based protocol for use in UANs. For best performance, the protocol parameters such as batch size and back-off window have to be optimally chosen. We show that an optimum batch size minimizes the total waiting time. Finally we compare our analysis results with experimental results obtained from a deployment of a small UAN.

***Index Terms*—Medium Access Control, Ad hoc Underwater Networks, Performance Analysis, MACA, Service Time Distribution**

I. Introduction

Underwater acoustic networks (UANs) spanning a few kilometers have many applications such as coastal monitoring, surveillance and surveying using Autonomous Underwater Vehicles (AUVs). Such networks are the focus of the Medium Access Control (MAC) protocol analysis presented in this paper. We believe that this is the first paper to present a full theoretical analysis of MACA based protocols for UANs, and to validate the analysis through field experiments.

A review of MAC protocols for UANs can be found in [1]. Small static UANs are best served by pure reservation protocols such as TDMA. If nodes are capable of communicating over multiple channels, a static cellular model where channels are spatially divided and allocated to cells according to a channel re-use pattern, is also a good option [2]. However, pure reservation based protocols are unable to meet the ad hoc demands of many underwater applications. For such applications, dynamic and ad hoc MAC protocols are necessary. One of the simplest dynamic and ad hoc protocol is Multiple Access with Collision Avoidance (MACA). The choice of MACA for UANs has been explored in detail in [3]. Variants of MACA are used in terrestrial networks such as 802.11 [4]. Protocols such as FAMA [5], DACAP [6] also are closely related to MACA. A simulation-driven comparative study of some of these protocols is presented in [7]. In [8], the authors present another evaluation of MACA-like protocols for UANs.

Several papers present analysis of MACA or its variants. In [9], FAMA, a close variant of MACA is analyzed. The authors assume that packet collisions are the only source of error, and ignore any loss due to noise (failure to detect or decode a packet due to noise). A three-way handshake with no acknowledgement is assumed. The paper lacks a saturated load analysis, service time distribution and a complete queuing analysis. In [4], the IEEE 802.11 DCF (which is also a variant of MACA) is analyzed. This paper also ignores packet detection and decoding losses. The analysis assumes a freezing back-off algorithm, rather than an optimal back-off window that we use in this paper. The service time equation derived is not expressed in closed form. In [10], a UAN oriented analysis of FAMA (called S-FAMA) is presented. The presented throughput performance of S-FAMA is low compared to the results we present in this paper. The expression for throughput does not capture the impact of batch size or back-off window size. The paper also lacks a saturated load throughput analysis similar to [4] and a complete queueing analysis. A more detailed comparison of the S-FAMA protocol with our work is presented in Section III-E.

The above papers do not analyze the queuing behavior of MACA based protocols. In [11], the authors present a queuing analysis of the 802.11 MAC. However, the analysis does not allow for detection and decoding losses, reliable delivery of packets across a link or transmission of a batch of data packets to minimize the effect of long propagation delays. The analysis is specific to 802.11 and its freezing-back-off model, and therefore not directly applicable to the optimal back-off window based protocol that we propose in this paper.

As we see, most of the analysis published to date focus on terrestrial networks and therefore ignore packet loss due to noise, and the long propagation delays encountered in underwater networks. The higher packet loss calls for link level acknowledgments. The long propagation delays call for sending a batch of packets for every RTS/CTS exchange (request-to-send/clear-to-send). The papers that address these issues only present limited analysis and focus on simulation results. A detailed analytical model and appropriate queuing analysis is missing from these papers. In this paper, we present a novel MACA-based protocol for use in UANs along with a detailed mathematical analysis of its performance. We derive performance bounds on the protocol through the use of

Manuscript received Apr 25, 2011, revised Jan 27, 2012, revised Nov 26, 2012, accepted on Jan 9, 2013.

S. Shahabudeen (e-mail: shirazs@gmail.com) is with NeST Software, India. M. Motani (e-mail: motani@nus.edu.sg) is with the Electrical & Computer Engineering Department, National University of Singapore. M. Chitre (e-mail: mandar@arl.nus.edu.sg) is with the Electrical & Computer Engineering Department and the ARL, National University of Singapore.

realistic models and parameters, and validate them through a combination of analysis, simulation and experiment.

The key contributions of this paper are summarized below:

- An accurate Markov chain based analytical model for a MACA based protocol for underwater ad hoc networks is developed.
- A batch data queuing analysis for reliable transfer for the protocol is developed. The analysis shows that an optimal batch size minimizes the total waiting time, whereas increasing the batch size arbitrarily maximizes the throughput.
- An ARQ protocol variation (Early-Multi-ACK) is proposed for efficient and reliable data transfer.
- Comparative sea trial results validate that the proposed protocol can indeed be implemented in a real network, and that its performance is similar to our predictions based on the analysis.

The analysis presented in this paper was influenced by some key ideas that have appeared in other papers. Specifically, the saturated load analysis was motivated by [4]. The Markov chain analysis to first find the expected service time for successful transmission was influenced by [12]. The service time distribution analysis by comparison with standard distributions was motivated by works such as [13]. The idea of dummy states used in service time distribution analysis can be found in papers such as [14].

Through a novel analysis, this paper presents new insights on the well-studied MACA protocol family, with primary focus towards UANs. The basic modeling is discussed in section II. Service model related measures such as expected service time and throughput are derived in section III and the service time probability distribution is derived in section IV. The analysis of queuing and total delay behavior is presented in section V. This is followed by further analysis on the optimum back-off window. Although the analysis presented can be applied to terrestrial wireless networks, our primary focus is to analyze the underwater MAC problem and therefore we do not present adaptation of the analysis to more general networks in this paper.

II. System Model

In this section we discuss the system model used in the paper, including the arrival and departure models, key model parameters, the protocol model being analyzed and the performance measures.

A. Input-output models

In the arrival (input) model we use, each data packet from network layer, or other layers above the data link layer (DLL), fits within a single DLL / physical layer packet. We consider both saturated load and Poisson arrivals. If the higher layer data size requires the usage of multiple DLL / physical layer packets, fragmentation and reassembly may be needed. However, we do not model fragmentation / reassembly in this paper.

For the service model, we adopt a packet train model where the DATA is sent in batches of size B with re-transmissions at the DLL (infinite retry model). For the retry mechanism, we propose some novel enhancements as detailed later in the paper.

B. Packet detection, error and collision model

Each packet has fixed length detection preamble at the start. Detection probability P_d is dependent on the nature of the preamble. Packet decoding probability P is determined by the bit error rate (BER) of the physical layer, the number of bits in the packet and the coding scheme. The probability that a packet is detected and decoded correctly k, is:

$$k = P_d P \tag{1}$$

Control and data packets may use different modulation, coding and packet length, as robustness is of key importance to control packets while data rate is of importance in data packets. To model this, we allow the decoding probability P_D of data packets to differ from that of control packets. Therefore the overall data packet success probability k_D is:

$$k_D = P_d P_D \tag{2}$$

Let the time duration (in seconds) of a control channel packet be L while that of a data packet be L_D. The maximum propagation delay is D. The number of nodes in the collision domain is N; we assume no hidden nodes [9]. 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 best viewed as the number of neighboring nodes that each node effectively contends with (section VI-G briefly outlines a rudimentary analysis for multi-hop networks).

C. MACA-based protocol model

The protocol is based on MACA using RTS/CTS (request-to-send/clear-to-send) exchange [15]. The basic model used is RTS/CTS/DATA-TRAIN/ACK. The transmitter sends RTS and the receiver sends back CTS. The transmitter then sends a batch of DATA packets (DATA-TRAIN). The receiver then sends a single acknowledgement (ACK) which indicates failed packets in the batch. Similar protocols with packet trains that employ ACKs after every packet (RTS/CTS/DATA/ACK/DATA/ACK…) are not efficient for UANs due to the two-way propagation delay overhead and thus we use only a single ACK at the end. We elaborate on the protocol further below.

In the RTS contention algorithm, a node starts with a uniform probability distributed back-off in a contention window W. When the back-off timer expires, a RTS is sent. Timer t_A is started when RTS transmission begins. If the timer expires before reception of CTS, 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 [4] whereas we use a constant window. Also, this protocol does not use Physical

Carrier Sense (PCS) whereas it is used in 802.11 (See VI-C for comments on PCS usage). All nodes use the same contention window W at any given time. The RTS and CTS specify the batch size and the VCS and DATA reception periods are computed at the nodes based on that.

The timers used to wait for CTS and ACK (t_A) are related to D and control packet time duration L to give enough time for the round trip delay as

$$t_A = 2D + 2L \tag{3}$$

To make reliable transfer more efficient, we propose two variations with regards to acknowledgments and retransmissions to handle failed data packets. After a batch of DATA is received, an ACK is sent by the receiver. In typically used retry models, if an ACK fails to reach the transmitter, the RTS/CTS based contention cycle and batch DATA transmission processes repeat. We introduce two enhancements to this retry process. Firstly, instead of sending one ACK packet, we send i ACK packets, a feature termed Multi-ACK. The second enhancement is as follows. When the sender of the DATA train does not receive the ACK, RTS is repeated with the same UID (unique identification number, incremented only for a RTS for a new packet train). The receiver sends back an ACK instead of CTS for the repeated RTS. Together with the Multi-ACK feature, we call this the Early-Multi-ACK model. The retry mechanism uses constant back-off with infinite retries (other options include exponential increase exponential decrease, maximum retries capped, etc.).

D. Performance measures

Queuing theory is commonly used in the modeling and analysis of wireless networks. Typically, the arrival process is modeled as Poisson distributed and the service time as exponentially distributed. A Markov chain analysis is then used to study the behavior the system. Important common metrics derived are service time distribution and its expected value, throughput efficiency, expected steady state queue length and expected total waiting time.

We define the mean packet service time s_p as the expected delay from the time a packet is intended for transmission (RTS contention starts) until it is successfully delivered, i.e., until the ACK (with retries) confirms successful reception of the specific packet. 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 successfully transmitted, i.e., until the first ACK is received for the batch. The above different definitions of s_p and s_b are important for the queuing analysis in Section V. In section III, we relate expected service times (s_p and s_b) to the network parameters as follows

$$s_b = f(N, D, L, L_D, B, k, k_D, W, t_A) \tag{4}$$
$$s_p = g(N, D, L, L_D, B, k, k_D, W, t_A) \tag{5}$$

Another important performance metric for reliable transfer is throughput, which we analyze in section III-C. In some papers on similar protocols in radio networks, this is termed as “saturation throughput” – the throughput of the network when the queue is saturated or always has data to transmit [4]. Such a measure is valid for file transfer applications. This 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 ($1/L_D$). B packets are sent as a batch in time s_b, and of these only k_D succeed on an average, due to decoding and detection losses. Thus, the normalized throughput T per node is

$$T = \frac{k_D B / s_b}{(1/L_D)} \tag{6}$$

In Section IV, simulations and numerical analysis examine the service time probability distribution. Once we characterize the service behavior with mean service times s_p and s_b and the service time CDF, other queuing metrics such as waiting time and queue length under non-saturated conditions (Poisson arrivals, etc) can be derived using queuing analysis (Section V). The total waiting time W_T includes the waiting time in the queue W_Q and the mean service time s_p per input packet, i.e., $W_T = W_Q + s_p$.

E. A brief note on simulations

The simulator used for this study is described in detail in [16]. In the simulation model, all nodes have data to send and each node sends data to one other recipient node. The simulator accurately models collision, decoding and detection errors, propagation delays, etc. The simulator has been enhanced to allow DATA and control packets to have different lengths and decoding probabilities (a limitation mentioned in [16]). It’s based on Omnet++ [17], an established discrete event simulation system. The nodes are randomly spread in an area whose dimensions are chosen to match the required maximum propagation delay D. The complete algorithm for the protocol discussed here has been implemented for both simulations and the sea-trials. The software system allows the same algorithm code to be used for simulations as well as sea-trials, i.e., there is no code porting required for sea-trials. This ensures that the simulation results and sea-trial results will have no artifacts due to possible porting differences and errors. Sea-trial related details are discussed in Section III-F. For sea-trials, key parameters are estimated and then plugged into the corresponding simulations and analysis models. For e.g., the probability of detection and decoding is estimated by sending a series of packets and recording the number of detected and successfully decoded packets.

III. Analysis of Expected Service Time and Throughput

In this section, we derive expressions for the expected service time and the saturated throughput. These closed form expressions closely match simulations and can be used for estimating protocol behavior and considerably reduce the need for simulations. The expected service time metrics are then used in Section V for queuing analysis.

A. Markov chain model for the protocol excluding retries

In [10], the authors show that the performance of RTS/CTS based protocols (Slotted-FAMA) improves with slotting. Based on this, a slotted model was chosen for our protocol. In line with the definition in [10], during RTS contention phase, the slot duration l is defined as

$$l = L + D \tag{7}$$

This allows for collisions to be contained within the slot boundaries. For $D \leq L$, packets transmitted in the same slot will at least partially collide. For example, for the NUS/ARL modem [16], [18], highly robust control packets have duration $L = 0.6$ seconds. Thus the model is very effective for D of about 0.6 seconds (900 m range). For $D \gg L$, packets in the same slot might not collide and the analysis is expected to give a conservative bound. For $D \gg L$, 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.

The protocol model is as described earlier in Section II-C. A node starts with a uniformly selected back-off time slot in the integer range $[1, W]$. The actual contention window time period is Wl. For simplicity of analysis, we assume that no collisions happen during the CTS period, assuming VCS starts due to RTS reception (results showed that this simplification did not have significant impact on the analytical predictions). So in our analysis model, CTS loss will only be due to decoding and packet detection probability. If the transmitter does not get CTS, it restarts the contention window for RTS. Any other node that 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. 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. This can be seen by noting that the probability of getting at least one packet after n packets are sent is $1 - (1 - k_D)^n$ which rapidly tends to 1 as n increases. Thus the contention cycle synchronization is maintained. This is similar to the NAV concept used in 802.11.

RTS packet transmissions are scheduled at the start of a time slot only; other response control packets are transmitted immediately to allow immediate VCS. DATA packets do not use slotting and there are no gaps between the DATA packets in a batch.

The protocol is represented using the main model in Fig. 1 and a supplementary model in Fig. 2. The main model accounts for the RTS/CTS process until a batch of DATA is transmitted. The supplementary model accounts for the ACK process. The absorbing state 6 in Fig. 1 and state 7 in Fig. 2 are for mathematical convenience and the protocol in actual operation does go back to state 1 and repeats the whole process after one cycle is complete. Circles with enclosed numbers are states. Transition probabilities are shown along the arrows.

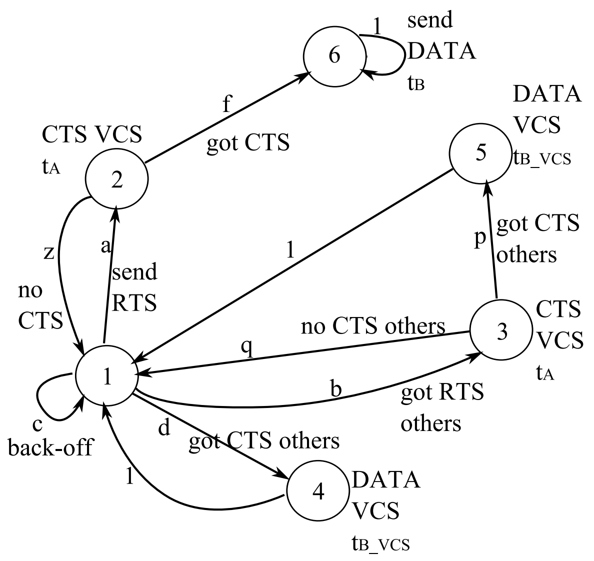

Fig. 1. Main Markov Chain for computing Expected Service Time

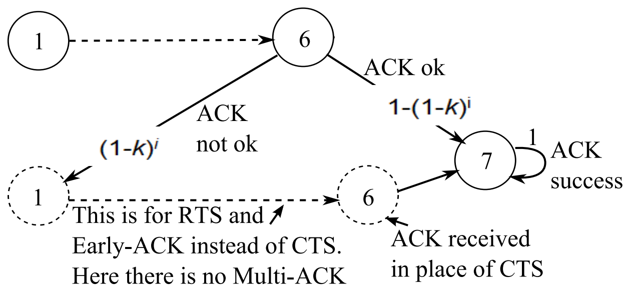
ACK ok
1-(1-k)^i
ACK not ok
(1-k)^i
1 ACK success
This is for RTS and Early-ACK instead of CTS. Here there is no Multi-ACK
ACK received in place of CTS

Fig. 2. Markov chain for Early-Multi-ACK

The duration spent in state 1 is 1, and for others states is t_B, t_{B_VCS}(described later in (10), (11)) or t_A as indicated. In the 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 $P(g, h)$.

The start of RTS contention cycle is at state 1. We model the probability of a node sending a RTS at the start of a new slot as $P(1, 2) = a = 2/(W + 1)$. This is because the expected value of the uniformly distributed contention window is $W' = (W + 1)/2$, and we use that as the expected value of a geometric process for transition (1,2) to satisfy Markov Chain requirements. 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 duration t_B (this is described later in (10)).

For notational convenience we shall define $\omega = 1/W'$. The probability that the RTS transmitted in a given slot has no collision from any other node is $(1 - \omega)^{N-1}$, i.e., no other node transmits a RTS in that 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 Fig. 1 as $P(2, 6) = f = k^2 (1 - \omega)^{N-1}$. If CTS is not successfully received, transition (2,1) happens as shown with probability $z = 1 - f$.

If a RTS is not sent (probability $1 - \omega$), 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 $y=(N-1)\omega(1-\omega)^{N-2}$ using same arguments as in last paragraph. And k being the RTS detection probability, the current node could receive a RTS from another node with probability ky . Thus, the transition (1,3) with $P(1,3)=b=(1-\omega)\,ky$ occurs as shown.

In state 3, it awaits CTS for time t_A. Thereafter if CTS is successful (probability $p=k^2$, since both RTS needs to be independently received by recipient and CTS received by current node), it goes to state 5 for batch VCS (for time t_{B_VCS}, described later in (11)), following which it goes back to state 1 with probability 1 immediately. CTS failure in state 3 with probability $q=1-k^2$ takes the system back to state 1.

If during back-off as described in last paragraph (probability $1-\omega$), 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 $y=(N-1)\omega(1-\omega)^{N-2}$. 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 $P(1,4)=d=(1-\omega)\,(1-k)k^2y$ as shown. In state 4, VCS for batch reception (for time t_{B_VCS}, described later in (11)) occurs and goes back to state 1 thereafter with probability 1. Note that for transitions (3,5) and (1,4), its possible that the intended recipient of the CTS may not receive it, but the neighbors who overheard any CTS honours VCS for batch transmission. This is a design choice for this protocol. 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 with $P(1,1)=c=1-a-b-d$.

Note that for states except state 1, where more than 1 unit of time is spent, we could model it using an expanded Markov chain as used later on in Section IV. But the above simplified Markov chain gives correct results if we account for the time in each state as done in (12) for example.

A Markov matrix $\mathbf{M}$ [19] represents this as follows using $P(a,b)$ as shown in Fig. 1. $\mathbf{Q}$ is the transient state matrix.

$$\mathbf{M}=\begin{bmatrix} c & a & b & d & 0 & 0\\ z & 0 & 0 & 0 & 0 & f\\ q & 0 & 0 & 0 & p & 0\\ 1 & 0 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix}\quad \mathbf{Q}=\begin{bmatrix} c & a & b & d & 0\\ z & 0 & 0 & 0 & 0\\ q & 0 & 0 & 0 & p\\ 1 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & 0\end{bmatrix} \tag{8}$$

The fundamental matrix $\mathbf{F}$ [19] is the $\mathbf{F}=(\mathbf{I}-\mathbf{Q})^{-1}$. Let $E(m,n)$ be the expected number of times the system is in 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 [19]:

$$E(1,n)=F_{1,n};\quad E(1,1)=\frac{1}{k^2}\frac{W'}{\left(\frac{W'-1}{W'}\right)^{N-1}}$$

$$E(1,2)=\frac{1}{k^2}\frac{1}{\left(\frac{W'-1}{W'}\right)^{N-1}};\quad E(1,3)=\frac{N-1}{k} \tag{9}$$

$$E(1,4)=(1-k)(N-1);\quad E(1,5)=k(N-1)$$

B. Enhanced retry mechanism

We now analyze handling failed data packets through ACK and re-transmission, as briefly discussed in Section II-C earlier. After the process from state 1 to 6 in Fig. 1, ACK is sent by the receiver. In typically used retry models, if ACK fails to reach, the stages from state 1 to 6 repeat until ACK is successfully received. We introduce two enhancements to this retry process. Firstly, instead of sending one ACK packet, we send i ACK packets. The probability of correctly receiving at least one of them is $1-(1-k)^i$. The additional time required for multiple ACKS is $(i-1)L$ and round trip time t_A is allowed for ACKs to be delivered. Thus the batch transmission time t_B is:

$$t_B=BL_D+t_A+(i-1)L \tag{10}$$

In the second enhancement as mentioned in Section II-C, when the sender of DATA train does not receive the ACK, RTS is repeated with the same UID (unique identification number, incremented only for a RTS for a new packet train). Receiver sends back ACK instead of CTS for such repeated RTS. This is shown in Fig. 2. The VCS delay t_{B_VCS} spent in states 4 and 5 in Fig. 1 is computed as follows. When using the Early-ACK feature, instead of receiving CTS, some of it could be Early-ACKs. Though there is no real batch transmission, let's for clarity use t_E as the time associated for a pseudo batch transmission in state 6 of such an Early-ACK cycle ($t_E=0$ of course). If we consider n batch transmissions of t_B over a long period of time among other nodes, there are thus $n(1-k)^i$ occurrences of t_E. The expected VCS time in states 4 and 5 in Fig. 1 is thus $t_{B\_VCS}=(nt_B+n(1-k)^it_E)/(n+n(1-k)^i)$. Since $t_E=0$,

$$t_{B\_VCS}=\frac{t_B}{(1+(1-k)^i)} \tag{11}$$

Let the time till successful reception of CTS from state 1 to state 6 of Fig. 1 be s_{CTS} (excluding the batch transmission time in state 6). We get,

$$\begin{aligned}s_{CTS}=(l)E(1,1)+t_AE(1,2)+t_AE(1,3)+\\ t_{B\_VCS}E(1,4)+t_{B\_VCS}E(1,5)\end{aligned} \tag{12}$$

Using (9), we can simplify s_{CTS} as $s_{CTS}=\gamma+(N-1)t_{B\_VCS}$ where γ is

$$\gamma=\frac{l}{k^2W'}\left(\frac{W'}{W'-1}\right)^N(W'^2+W'-2)+\frac{2(N-1)l}{k} \tag{13}$$

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) as shown in Fig. 2. Based on Fig. 2, the total batch service time s_b from state 1 to state 7 (from RTS until Early-Multi-ACK) can now be computed using expected time from state 1 to 6 (including batch transmission) and additional time s_{CTS} for Early-Multi-ACK as follows

$$\begin{aligned}s_b&=s_{CTS}+t_B+(1-k)^is_{CTS}\\ &=\left(1+(1-k)^i\right)\gamma+Nt_B\end{aligned} \tag{14}$$

Equation (14) shows the average time for batch transmission from the perspective of a single node in a group of N nodes. The factor Nt_B has an intuitive appeal, since on average each node should get a turn to send DATA after $N-1$ batch transmissions by other nodes. A particular batch comprises of packets to be resend (those in the previous batch that did not get across), and new packets. In the Early-ACK model, such a specific batch (with a specific UID) is transmitted only once and the service time s_b refers to the time taken to send that. If the ACK is lost, the specific batch never gets resend, as an Early-ACK will be send in response to the repeat RTS. If the ACK is not lost, the next batch will be formed with a new UID and a RTS will be send for that. This new batch may even comprise entirely of old DATA packets if the ACK indicated that no packets made it previously, and even then it will still be considered as a new batch, with a new UID and another service time s_b applies to it.

An important observation is that in this protocol the contention process and the variable time associated with it (not including batch data transmission) depends only on control packet duration and its probability of success. It does not depend on individual DATA packet duration or its success probability. Total data packet transmission duration is captured though the term t_B in (14) and it depends on the product BL_D.

The following are some key comments regarding ACKs and the ARQ mechanism used in the protocol variants in this paper:

- ACK will be sent if any one packet in the train gets through. For simplicity, we assume that the receiver will get at least one packet and hence it will send back an ACK with probability 1 (probability of getting at least one packet is $1-(1-k_D)^B$. E.g. for $P_d = P_D = 0.9$ and $B = 5$, P (get at least one packet) = 0.9998).
- During the Early-ACK phase, there is no Multi-ACK, since in this phase RTS/CTS exchanges are potentially in progress among other nodes and it will be a protocol violation. We can only send one ACK packet, which is of the same size and coding type as a CTS packet, for correct reception during this contention phase.
- The above reliability mechanisms are related to general ARQ strategies. The analysis here can be considered an ideal form of retry based reliability. ACK is assumed to convey complete information on lost packets and re-transmissions happen only on lost packets. Other standard and practical forms of ARQ such as the Selective Repeat ARQ in TCP/IP, etc may have limitations on amount of information that can be contained in ACK. They would be less efficient since re-transmissions may potentially happen for packets previously received. Thus the results here can be viewed as an upper bound on reliable performance using batch mode service as described in this paper.
- We can find the optimum number of ACKs i as discussed in Section VI-F.

C. Expected throughput for reliable transfer

Using (6) and (14), we show the throughput in Fig. 3 for one scenario from both analysis and simulations (this is the network throughput, i.e., NT). The analytical results match the simulation results well. In Fig. 4, we show simulation results for the unslotted model to ascertain the effectiveness of slotting as discussed in Section III-A. As seen, there is little difference from the slotted model for the various simulations undertaken in this study. Based on this finding, the sea-trials discussed in Section III-F do not use slotting. This result also uses longer DATA packets with lower probability of success and larger D.

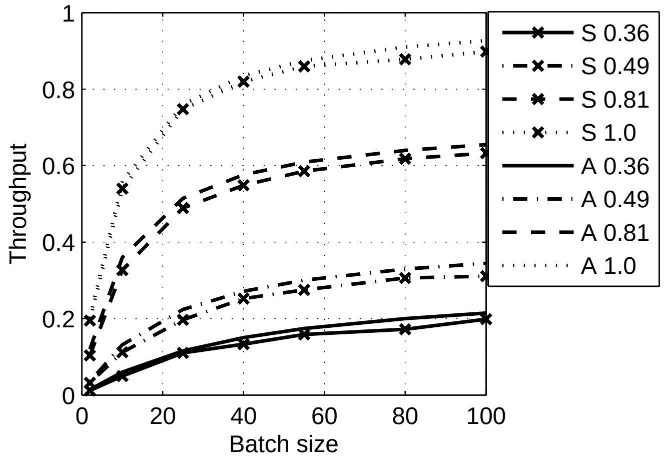

Fig. 3. Throughput vs. batch size for k = 0.36, 0.49, 0.81 and 1.0, k_D = 0.3, 0.42, 0.72 and 1.0, $L = 0.5s, L_D = 1.0s, N = 7, W = 17, i = 3, D = 0.5$, simulations (S), analysis (A)

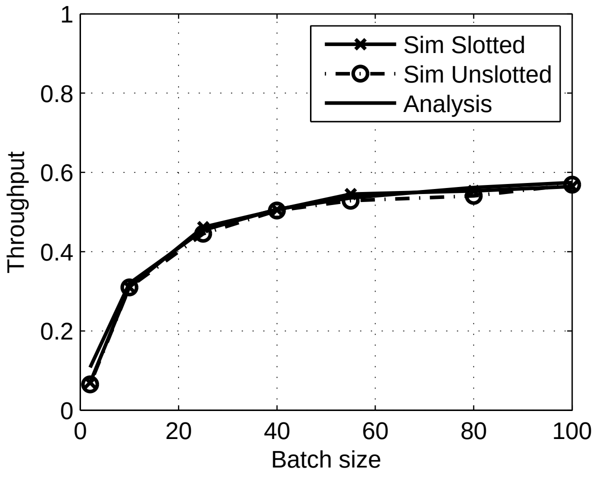

Fig. 4. Throughput vs. batch size for $[L = 0.5s, L_D = 1.5s, k = 0.81, k_D = 0.63, N = 7, W = 17, D = 1.0, i = 3]$

The throughput has a saturation type behavior at higher batch sizes. Using (6) and (14), saturation throughput behavior is $\lim_{B\to\infty} T = \frac{k_D}{N}$, which is the theoretical upper bound for one-way transmissions (when the combined detection and decoding success probability is k_D). To compare, the performance of the standard ACK model (without Early-Multi-ACK) is $\bar{s_b} = \frac{1}{k}(\gamma + N\bar{t_B})$ (see Appendix A for details). The corresponding asymptotic throughput is $\lim_{B\to\infty} T = \frac{kk_D}{N}$. This performance is lower by a fraction k of that of the enhanced ARQ model proposed in this paper.

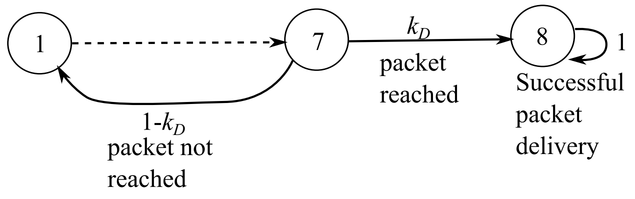

Fig. 5. Markov chain to compute s_p

D. Expected packet service time s_p

The average reliable delivery service time s_p is measured on a single packet within a batch. ACK indicating error packet determines if any re-transmissions happen in the next round as shown in Fig. 5. We can then get s_p as

$$s_p = \frac{1}{k_D} s_b \tag{15}$$

E. Comparison with previous analyses

For $k = k_D = 1$ in Fig. 3, we look at the collision-limited performance of the protocol variants. This is the scenario used in most papers on UANs (e.g., [20]), i.e., only collisions are assumed to cause losses. Based on Section III-C, T will converge asymptotically to $1/N$ using the enhanced retry mechanism if $k = k_D = 1$. Most previous analysis and simulation do not consider detection (P_d) and decoding (P) based packet losses. These are important loss factors in UANs. By taking into account both P_d and P in our analysis, we believe that the results here provide a strong basis for MACA-based protocol performance analysis for underwater applications. We also highlight that our protocol performance analysis includes retry based reliability.

We also note some key differences with S-FAMA [10] as it is a very similar protocol. Features such as Early-ACK, Multi-ACK are absent in S-FAMA. S-FAMA sends all packets including DATA only at the start of a slot, whereas MACA-EA sends only RTS at the beginning of a slot. Longer contiguous DATA transmission in S-FAMA is typically achieved through the use of single long DATA packet (simulations use DATA packets that are thirty times longer than the control packets), where as packet trains with short DATA packets are used in MACA-EA. DATA packet trains are also considered in S-FAMA, but the acknowledgement mechanism uses one ACK per DATA packet, where as MACA-EA uses a single ACK at the end of a batch. The best average throughput per node is shown to be about 0.022 at a range of about 1.9 km, where the average number of neighbors is shown to be 4. It is low as compared to the best per node throughput performance of MACA-EA (shown to approach $1/N$ in the absence of detection errors, so predicted to be about 0.25 for 4 neighbors). However, the results they have presented are not for saturated traffic and therefore may not be directly compared with our results.

F. Sea trial results

In this section we briefly summarize the key findings from acoustic modem sea trial results. Trial details can be found in [16]. Some of the environmental parameters were as follows: sound speed profile – flat with sound speed of 1540 m/s, water temperature – about 30 deg C, salinity – about 35 ppm. Key parameters include number of nodes $N = 3$, maximum distance between nodes D of about 400–500 m, minimum separations of about 200 m (Fig. 6), un-slotted protocol implementation, saturated load (file transfer) application. Multiple tests were performed at each parameter setting as permitted by allocated time during the sea-trial. Note that the same MAC protocol C code is used in the modem and the simulator through a unified simulator and modem software interface. We used a maximum delay D of 0.4 seconds as an upper bound to trial distances, contention time window Wl = 10 seconds and batch sizes B = 5, 10 and 40. A saturated traffic model is used. In the sea trials, estimated $P_d = 1$ and $P = P_D = 0.9$. This trial also used $i = 1$, i.e., only one ACK, as the trials were performed before the multi-ACK feature was introduced in the protocol. The modem used for the trial uses a Power Amplifier (PA) for transmission which takes about 300 ms to settle after being turned on. We term this delay as t_{PA}. When PA is turned on, no reception is possible. For batch transmission, PA is turned on/off only at the start and end of the batch. This PA behavior is captured in the simulations. Thus for the analysis, even though the actual packet duration (highly robust control or data packets in the NUS/ARL modem) is 0.6 seconds, an effective time period of 0.9 seconds has to be considered in the contention part, i.e., $L = 0.9$. During batch transmission, since PA is turned on/off only at the beginning and end, we define a different data packet length $L_D = 0.6$. Thus we modify (10), as $t_B = BL_D + t_A + (i-1)L + t_{PA}$. The results are shown in Fig. 6. We find that the analysis closely matches with simulation results as well as the sea trial results. The sea trials thus give strong validation for both simulation and the analysis. It should be noted that the performance at sea was slightly worse than the predictions, due to some of the simplifying assumptions made in the analysis and simulation models. During the sea trials, image files (see Fig. 6) were transferred using this protocol with Early-ACK mechanism to retransmit failed packets.

In the system used for the experiments, control packets (RTS, CTS, ACK) were designed to have a small payload capacity to carry PHY tuning parameters, power control, and other network related information. Although many of these features were not used in the experiments, they were implemented for future use. Each of the DATA packets were also of the same modulation, coding and length, and used this payload capacity to carry application data. Section VI-D discusses the use of longer DATA packets.

Tight time synchronization was not used in experiments due to practical limitations, and hence the implementation should be considered as un-slotted. As discussed in Section III-C, the difference between slotted and un-slotted models was found to be marginal (see Fig. 4). Thus the slotted analysis used here still presents a reasonable approximation. It is certainly

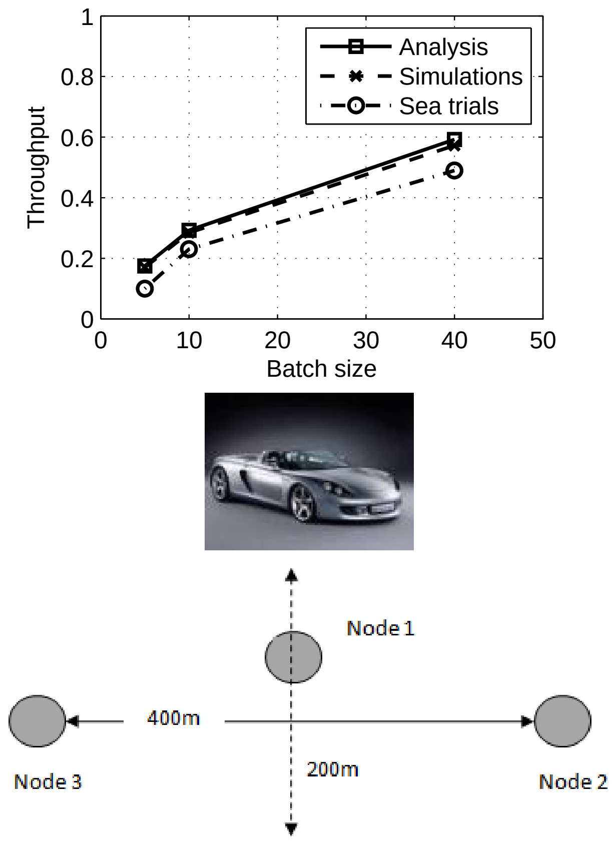

Fig. 6. Analysis comparison with sea trials and simulations. Parameters: $N = 3, D = 0.4s, W = 10, P_d = 1, P = 0.9, i = 1, L = 0.9s, L_d = 0.6s$. Also shown is a sample image that was transferred between modems in a recent sea trial. This file transfer used the MACA based protocol with the Early-ACK retry mechanism. The estimated relative node separations are also shown.

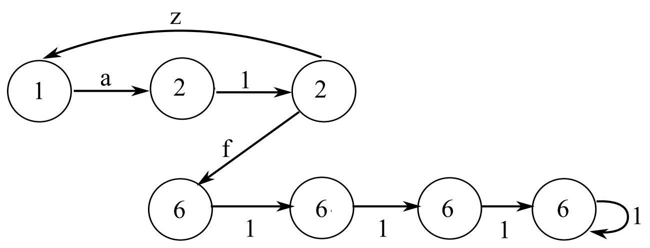

Fig. 7. Part of Markov chain with dummy states for computing service time distribution

possible to add slotting to UAN experiments through the use of accurate synchronized clocks, but the added complexity may not be warranted as the performance gain may be marginal.

IV. Service Time Distribution Analysis

In this section, we analytically and numerically compute the service time distribution. The probability distribution is important to ascertain the validity of the assumptions used in queuing analysis in the next section.

Dummy states are introduced for the non-unity delay states with time delays t_A (states 2 and 3) and t_B or t_{B_VCS}(states 4, 5, 6) in Fig. 1. To illustrate this concept, a part of Fig. 1 is shown in Fig. 7, where transition probabilities between dummy states are 1. The example shows t_A of 2s in state 2 and t_B of 4s in state 6. The packet can reach the final absorbing state (complete transmission) through many paths. Each path will have a finite number of steps and the number of steps directly relates to the time (or the number of slots) taken to reach the absorbing state. This is the service time s_b for that path. Using Markov chain properties, we can find the expected time over all paths to reach final state.

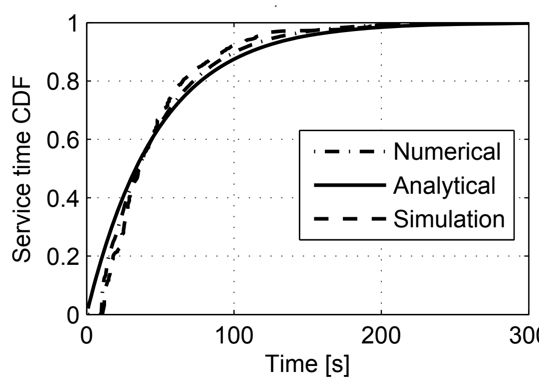

Fig. 8. Service time CDF. $B = 10, L = L_D = 0.5, N = 3, D = 0.5, W = 4, i = 3, k = k_D = 0.81$. "Analytical" curve uses Exponential fit.

We first implement the Markov chain shown in Fig. 1 in Matlab using dummy states as mentioned above. For illustration, we show one example in Appendix B. The transition matrix $\mathbf{M}$ is shown in (36). Note that this matrix is only one realization for a certain parameter set. For the experiments below where batch size B is varied, which effectively varies t_B, the number of total states and hence the transition matrix $\mathbf{M}$ is varying. This is also the case for variations that cause t_A, etc to change. These matrices are dynamically formed by a Matlab program. Following that, we use $\mathbf{M}$ together with the additional transitions in Fig. 2 for the retry scheme to form the complete Markov chain $\mathbf{M_R}$ as shown in Appendix B.

We see that the matrix $\mathbf{M_R}$ is in the canonical form of an absorbing Markov chain with one absorbing state. Using standard Markov chain theory, we can compute the probability of being in the final state after starting from the initial state in n-steps. $\mathbf{M_R}_{i,j}^{(n)}$ of the matrix $\mathbf{M}^{(n)}$ gives the probability that the Markov chain, starting in state s_i, will be in state s_j after n steps. The CDF for a particular scenario is shown in Fig. 8 with batch size $B = 10$ (In Appendix B, a smaller batch size is used to keep the illustrative matrix small). Simulation is also used to estimate the distribution nature and it shows excellent match to the numerical analysis. In queuing theory, exponentially distributed service time is commonly used. To compare, exponential distribution curve is shown, which uses the analytical mean service time from (14). The numerically computed (and the simulated) CDF shows close similarity to an exponential distribution behavior.

Note that these protocols have a finite minimum service time due to the handshaking till an ACK is received even if the sequence RTS/CTS/DATA/ACK succeeds in one round. Such a larger batch size example is shown in Fig. 9. The numerical computation (based on the analytical model in this paper) matches the simulation results reasonably well.

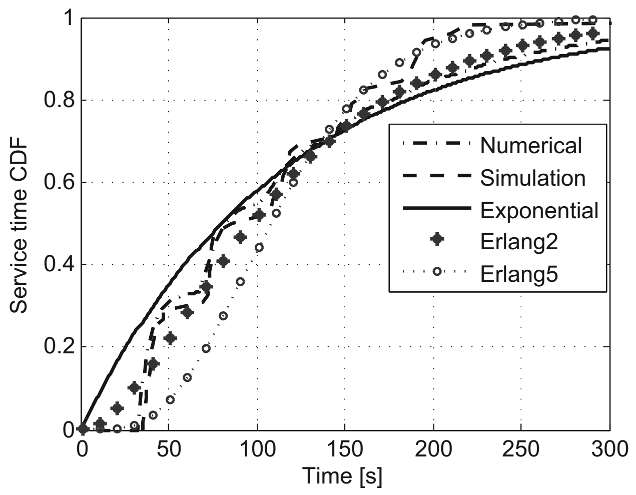

Fig. 9. Service time CDF. $B = 55, L = L_D = 0.5, N = 3, D = 0.5, W = 4, i = 3, k = k_D = 0.81$.

Although the numerically derived CDF somewhat deviates from an exponential distribution, it provides a reasonable approximation. We also investigated Erlang distributions which offer a more general class of distributions that offer a better fit in some cases. The Erlang-K distribution is formulated as a sum of K exponential distributions. The maximum likelihood estimate (MLE) for the Erlang-K is the K that minimizes $K + Klog(A/M) - (K+1)log(K) + log(K!)$, where A is the arithmetic mean of the data, and M is the geometric mean of the data [21]. Using this, $K = 2$ was found to be optimal in multiple sets of service time data with different batch sizes, and a plot is shown in Fig. 9. An Erlang-K distribution with $K = 5$ also shown for comparison. Based on this comparative study, we assume exponential service time distribution to provide a basic approximation for further analysis, and use Erlang-2 ($K = 2$) as a slightly better approximation.

V. Queuing Analysis

The extremely low data rates of UANs are a bottleneck in many co-operative underwater missions and the data generated may be in excess of the capacity, thereby treatable as saturated traffic. File transfer applications also provide an example of saturated traffic scenarios. Metrics such as throughput in such cases can be analyzed as discussed in Section III. In some scenarios, data generation rates could be lower than system capacity and can be treated as unsaturated. In this section we address this latter case.

We consider Poisson arrivals with rate λ as in standard queuing theory analysis. The service time distribution can be modeled as exponential or Erlang-K with $K = 2$ (Erlang-2). Thus the system is modeled as either M/M^B/1 queue [19] with exponential batch service or M/E_2^B/1 queue [22] with Erlang-2 batch service for queuing delay analysis.

A. Unsaturated queuing analysis

All nodes contend after each batch transmission as discussed in Section III-A. The long durations between batch transmission opportunities for UAN nodes make it inefficient for a node not to contend until it has DATA to send. So in our protocol, a node will restart contention immediately after the current cycle even if there are no packets in the buffer. The expected time till it gets a successful CTS according to (14) is after $(N-1)$ other batch transmissions and contention cycles. During this period a node gets to accumulate DATA from Poisson arrivals in the queue. When CTS arrives, a node starts sending DATA in the queue, while remaining open to new arrivals until the allowed batch transmission time is complete, i.e., new arrivals immediately enter service up to limit B and finish with others. In addition, there are failed DATA packets to be retried. Thus in our model, the number of contending nodes is not dependent on the arrival rate at each node. The process consists of contention cycles followed by batch transmission of DATA packets. The validity of this pre-emptive model is evaluated in detail in Section VI-A.

The contention process consists of RTS and CTS exchanges between N nodes as described earlier and its average duration or its statistical distribution is independent of the ensuing DATA batch transmission. The DATA batch transmission is for a fixed duration and the total service time is the sum of the variable contention period and the fixed DATA transmission period. The definition of mean batch service time s_b is 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. Even if no DATA is available to be sent when CTS is received, the transmitter and receiver as well as all the neighbors diligently execute the batch transmission phase. The transmitting node continues to remain open to arrivals, and transmits the packets if they arrive within the allocated batch transmission period. ACK will be sent by the receiver indicating the number of packets received (even if no packets are successfully received) after the batch transmission period is over. Thus as long as the number of nodes in the contention phase remains the same (apart from environmental and system factors), the batch service time s_b does not change. This is the base metric for much of the analysis (including throughput) as seen earlier. This implies that the service time distribution is independent of DATA content in the batch transmission phase (i.e., there may or may not be sufficient packets in queue to fill the batch size B). The exponential (or Erlang-2) nature is fundamentally a result of the contention process. And hence we can use it as the basis for the queueing analysis for the non-saturated case. The exponential (Erlang-2) service time remains valid as long as all nodes contend during each contention cycle, which is the protocol model used. In the queueing system, the server keeps serving, whether or not there are packets in the queue, and is unaffected by arrivals or the contents of the batch transmission phase itself.

B. Expected waiting time

Expected values for steady state queuing length and waiting times can now be derived using queuing theory for batch service. First we look at exponential service time. The probability generating function of the steady state probability distribution $P(z)$ for M/M^B/1 (can be inferred from results in [19]) is

$P(z) = (1-r_0)/(1-r_0 z)$. We define service rate μ_b as $1/s_b$. The constant r_0 is the root of the characteristic equation of M/M^B/1 [19] as shown below.

$$\left(\mu_b z^{B+1} - (\lambda + \mu_b) z + \lambda\right) p_n = 0; n \geq 0 \quad (16)$$

The root r_o needs to be found numerically except for small values of B. When $B = 1$, (16) becomes quadratic. Roots of (16) can analytically be found up to $B = 4$ using Matlab's symbolic math tool box, but expressions are cumbersome. For $B >= 5$, analytical roots are not always obtainable as is generally known in polynomial theory.

We can compute required moments such as expected queue length and waiting time as follows. Expected system total length L_T (includes the packets in service)

$$L_T = P'(z)|_{z=1} = \frac{r_0}{1-r_0} \quad (17)$$

For Erlang-K, the queue length can be found as [22]

$$L_T = \sum_{B}^{B+K-1} (z_j - 1)^{-1} \quad (18)$$

where z_j are the roots of

$$z^B - \left(1 + \frac{\lambda}{\mu_b}(1-z)\right)^{-K} = 0 \quad (19)$$

Using Little's Law [19], expected waiting time W_Q (excluding service) can be found as follows for either Exponential or Erlang-K (s_b from (14))

$$W_Q = \frac{L_T}{\lambda} - s_b \quad (20)$$

Total waiting time W_T (the sum of queuing time W_Q and service time of one packet s_p is then (s_p from (15))

$$W_T = \frac{L_T}{\lambda} - s_b + s_p \quad (21)$$

The average intake batch size is reduced to Bk_D, since retries will occupy $(1 - k_D)B$ on average for every batch, this modified batch size should be used in (16) and (19).

C. Waiting time variation with batch size

We look at one example of how W_T varies with batch size in Fig. 10, with parameters $L = L_D = 0.5, N = 3, D = 0.5, W = 4, i = 3, k = k_D = 0.81$ and $\lambda = 0.05$. Simulations match the analytical results well and validate the accuracy of the analytical model. As seen, in this case the exponential and the Erlang-2 results are very similar (a small difference seen at $B < 10$). We also look at another example in Fig. 11 with a different set of parameters $L \neq L_D$, $k \neq k_D$ and other differences as shown.

This shows that there is potentially an optimum batch size that minimizes the total waiting time. Saturated throughput analysis in Section III-C showed that by increasing the batch size arbitrarily, high levels of throughput can be achieved, but it does not give the cost in terms of total delay encountered for Poisson arrivals. Thus for Poisson arrivals, it's best to optimize the batch size to minimize the waiting time. This is a key observation in this paper. Using the analytical model, we look at how different arrival rates and the number of nodes impact the optimum batch size in Fig. 12. We can see the optimum batch size increases with decreased arrival delay (increasing λ) and also with the number of nodes.

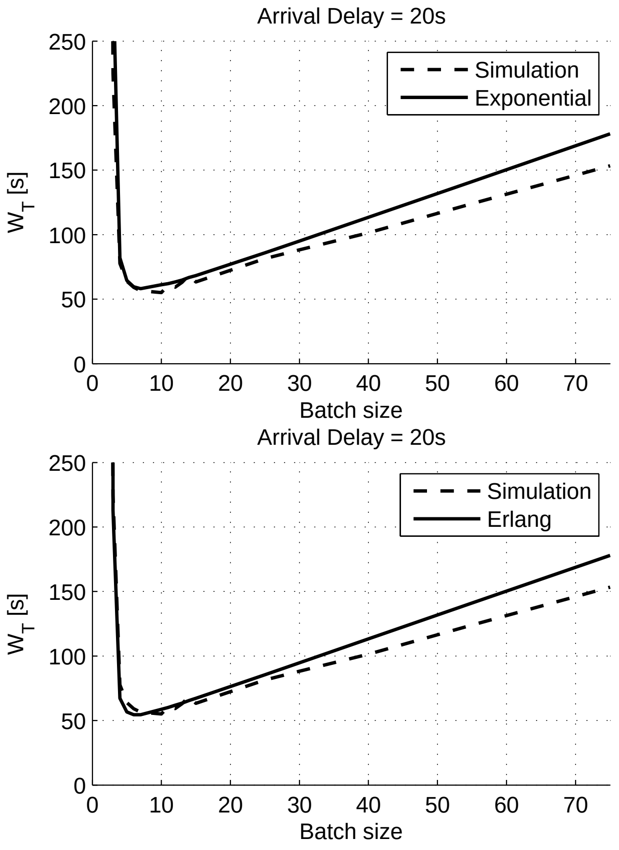

Fig. 10. W_T (in seconds) vs. Batch size (B), parameters $L = L_D = 0.5, N = 3, D = 0.5, W = 4, i = 3, k = k_D = 0.81$, $\lambda = 0.05$

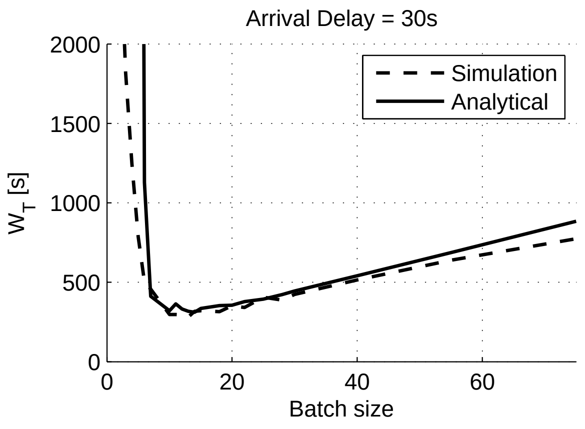

Fig. 11. W_T (in seconds) vs. Batch size (B), $N = 7, D = 0.5s, L = 0.5s, L_D = 1.0s, k = 0.81, k_D = 0.72, W = 17, i = 1, \lambda = 0.033$

We present analytical insights into the optimum batch size below using the exponential model. Define α and β as follows

$$\alpha = \left(1 + (1-k)^i\right)\gamma + N\left(t_A + (i-1)l\right); \quad \beta = NL_D \quad (22)$$

Using α, β we can simplify (14) as

$$s_b = \alpha + \beta B \quad (23)$$

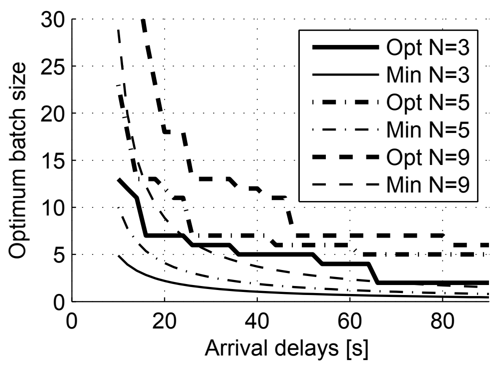

Fig. 12. Variation of optimum ('Opt') and minimum ('Min') batch size with the number of nodes N and arrival delays, parameters $L = L_D = 0.5, D = 0.5, W = 4, i = 3, k = k_D = 0.81$

Since on average $k_D B$ packets are transmitted in time s_b, for a stable system, the arrival rate must satisfy $\lambda < \frac{k_D B}{s_b}$. Therefore, we get $\lambda_{max} = \max(\lambda)$ for given B as follows

$$\lambda < \frac{k_D B}{\alpha + \beta B} = \lambda_{max} \quad (24)$$

and $B_{min} = \min(B)$ for a given λ as follows (shown along with the optimum in Fig. 12)

$$B > \frac{\alpha}{\frac{k_D}{\lambda} - \beta} = B_{min} \quad (25)$$

From (24), we can see that $B_{min} = 1$ if

$$\lambda < \frac{k_D}{\alpha + \beta} \quad (26)$$

For such λ, from (16) for $B = 1$ (non-batch mode use of the protocol), using (23) we obtain $r_0 = \frac{\lambda}{\mu_b} = \lambda(\alpha + \beta)$ and the corresponding waiting time relationship to λ can be shown to be as follows based on (21) (s_p from (15))

$$W_T \quad = \quad \frac{1}{k_D}(\alpha + \beta) + \frac{\lambda(\alpha + \beta)^2}{1 - \lambda(\alpha + \beta)} \quad (27)$$

Similarly, for $B \gg 1$ in (16), we can use an approximation $r_o = \lambda/(\lambda + \mu_b)$ and the corresponding waiting time from (21) can be shown to be as follows, which is independent of λ

$$W_{T\_A} = \frac{1}{k_D}(\alpha + \beta B) \quad (28)$$

The independence of above waiting time from λ occurs for such batch sizes, because every arrival gets served in the very next batch transmission for very high B. We illustrate the behavior in Fig. 13 where B_{min} (25) and the asymptotic lower bound (28) is indicated (the latter as a dash-dot line). This general behavior was observed in all numerical and simulation results. If W_T for $B = 2$ (based on (28)) is higher than W_T for $B = 1$ (based on (27)), we can infer that for

$\lambda < \beta/(k_D(\alpha + \beta)^2 + \beta(\alpha + \beta))$, there will be no such optimum behavior (presence of a minimum for W_T as illustrated in Fig. 13 or the specific example in Fig. 10) and W_T monotonically increases with B. For such small λ, $B = 1$ is the optimum batch size. For larger λ and still within the upper limit specified by (26), there may be an optimum batch

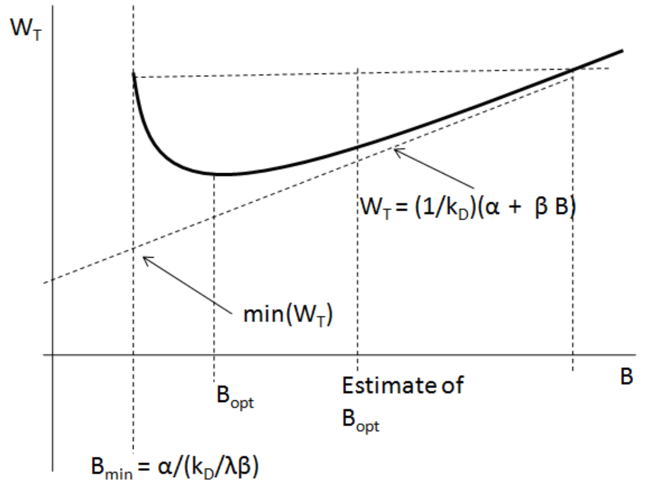

Fig. 13. Waiting time behavior illustration – the solid curve is W_T. Other key characteristics are as indicated.

size behavior similar to that seen in Fig. 10. We provide a simple engineering approximation of this optimum batch size $\overline{B_{opt}}$ as follows since an analytical solution is intractable. A near-optimum batch size is obtained from the simple graphical construction shown in Fig. 13; $\overline{B_{opt}}$ chosen as mid-point of B_{min} and B_x as shown.

$$\overline{B_{opt}} = 1 + \frac{k_D \lambda(\alpha + \beta)^2}{2\beta\,(1 - \lambda(\alpha + \beta))} \quad (29)$$

For $k_D/(\alpha + \beta) < \lambda < k_D B/(\alpha + \beta B)$, $B_{min} > 1$ and the optimum batch size needs to be estimated using numerical computation as described in previous section. A general lower bound $\underline{W_T}$ can be found using B_{min} (25) in the asymptotic relationship of (28) as indicated in Fig. 13.

$$\underline{W_T} = \frac{\alpha}{k_D} + \frac{\beta}{k_D}\left(\frac{\alpha\lambda}{k_D - \beta\lambda}\right) \quad (30)$$

VI. Discussion

A. Pre-emptive contention

If a certain node is inactive and has no DATA to send for extended periods of time, MACA-EA protocol does not advocate that it keeps contending. In our analysis, that node will not be part of the equation. Only active nodes are considered in counting N, and by definition they are active only if they have DATA to send according to the specified arrival rate, assumed same for all nodes in this analysis. Since it takes a finite time t after a batch transmission before the next successful contention, the average queue size will be λt during that period alone. The extreme case of having no packets may occur with a small probability in this Poisson arrival model. Even if there is one packet to transmit, the server attempts to be fair to it by successfully contending to transmit it. In UANs, the opportunity to transmit may be very infrequent due to losses and low data rate, and hence for very active missions, it was considered desirable to proceed with contention even if there were no packets at the start of the contention. The expectation is that by the time contention is completed, there may be non-zero packets to deliver. Adaptive batch sizes can

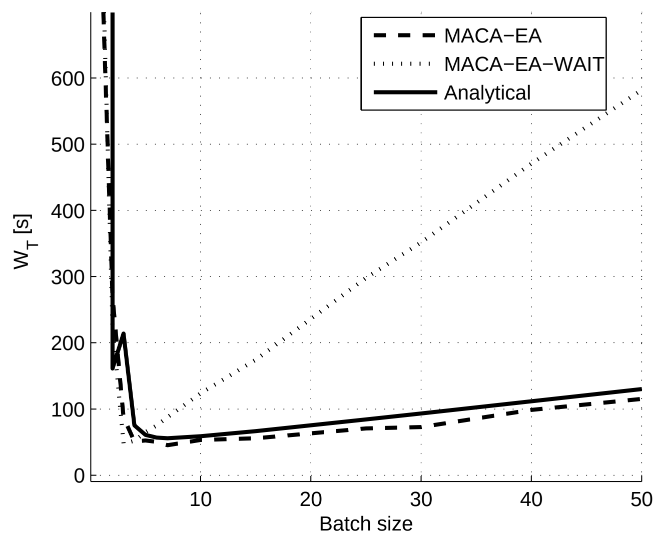

Fig. 14. Comparison with MACA-EA-WAIT, arrival delay = 20s. Parameters: $L = L_D = 0.5, D = 0.5, N = 3, k = k_D = 0.81, i = 3$.

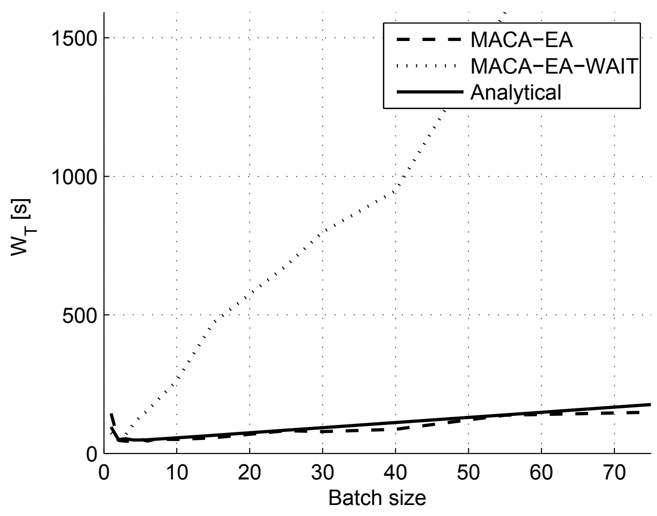

Fig. 15. Comparison with MACA-EA-WAIT, arrival delay = 50s. Parameters: $L = L_D = 0.5, D = 0.5, N = 3, k = k_D = 0.81, i = 3$.

be useful in such scenarios for a protocol in real applications. Adaptive batch size analysis is not undertaken in this paper. One of the potential scenarios is an AUV mission where there are a small number of nodes in a single collision domain, all actively exchanging data to each other. TDMA could be used for such a scenario. The motivation in moving from TDMA to MACA is to eliminate the extreme dependence on time synchronization and thereby increasing robustness, and to add ad hoc capability (nodes can arrive depart easily). In MACA-EA, unlike TDMA, we can adaptively stop using bandwidth if there is nothing to send for extended periods of time.

It should be noted that after successful contention (CTS received), if there is still no DATA, the batch transmission period will be honored by all nodes based on VCS principle. ACK will not be sent. Note that such occurrences will be related to packet arrival rate, and for lower rates, the optimum batch size also will be low, and $B = 1$ for λ below a threshold as discussed in Section V-C. So the wasted batch transmission time is not significant. Nevertheless, it may seem that the preemptive contention used by MACA-EA is not ideal. Therefore to test its merit, MACA-EA is compared with MACA-EA-WAIT, a variant in which the protocol waits for B packets to be in the buffer before contention starts. Two sample results are shown in Fig. 14 and Fig. 15.

The results indicate that the lowest waiting time for both MACA-EA and MACA-EA-WAIT are comparable. MACA-EA-WAIT has a slightly lower optimum batch size at which the minimum occurs. For high arrival rate (saturation throughput), MACA-EA-WAIT waiting time increases sharply with increasing batch size. As larger batch sizes give better throughput efficiency, the MACA-EA protocol can give reasonable throughput efficiency for saturated load and at the same time, reasonable waiting times for Poisson arrivals. Thus the preemptive contention model has an overall positive effect on the proposed protocol.

The reasons for the behavior of MACA-EA-WAIT are as follows:

- When contention starts only after B pkts are in queue, new packets that arrive between contention start and successful batch transmission need to wait till next turn, which will be on average $N - 1$ turns later.
- Packets that need to be retried cannot be tried immediately after the current transmission, but need to wait for buffer to get full.
- For higher batch sizes, the time to wait for buffer to get filled increases proportionally, and the combination of the above two factors seem to outweigh some of the gains that is possible by abstaining from contention.
- Contention and associated collisions is only one of the loss mechanisms. Packet detection and decoding losses also contribute greatly to the behaviors. In UANs, the latter could even dominate collision losses in small networks. (Note that analysis for 802.11, etc typically focuses only on collisions, and this factor is ignored in typical UAN MAC analysis).

B. Optimum RTS window

The optimum RTS window for throughput maximization can be derived as follows. Using X and U to represent terms which has no W' in the service time equation (14), $s_b = X(1/W')((W'/(W'-1))^N(W'^2 + W' - 2) + U$. From (6), T can be written as

$T = k_D B L_D/(X(1/W')(W'/(W'-1))^N(W'^2+W'-2)+U)$. Using $\partial T/\partial W' = 0$, we get the optimum for positive W' as

$$W' = \frac{1}{2}\left(N + \sqrt{N^2 + 8N - 8}\right) \qquad (31)$$

The approximation $W' \approx N + 2$ can be used for $N \gg 1$. The optimum contention window size is thus $W = 2W' - 1 \approx 2N + 3$. We observe how throughput changes with W in Fig. 16. Different lines represent different batch sizes. For $N = 10$ and other parameters as in Section III-C, Fig. 16 shows the throughput using the analytical formula and the optimum $W \approx 2N + 3$ can be seen. We see that varying batch size has no effect on optimum RTS window and it

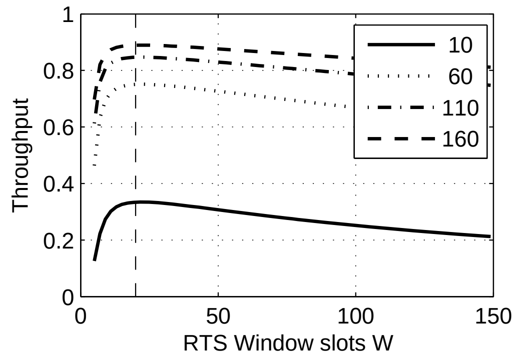

Fig. 16. Variation of network throughput with W (legend shows batch size B), $N = 10$, other parameters: $L = L_D = 0.5, D = 0.5, k = k_D = 0.81, i = 3$.

depends only on the number of nodes N, i.e., on neighborhood density. This result helps the MAC to dynamically select an optimum back-off window based on its estimate of the number of neighboring nodes (estimated based on the received packets over time since each packet has the sender identity) and eliminates the need for freezing back-off as employed in 802.11.

C. Physical carrier sense

A variation where PCS is used at the end of RTS phase is possible as in Bianchi's [4] model. We can assume that a transceiver is in receiving mode unless a transmission is made. If the back-off counter expires at a time when a reception was in progress, we can consider it as a PCS that prompt a back-off. Our simulations show that with the protocol variations and the parameters used here, PCS just before sending RTS provides only a small gain on top of VCS based on RTS, CTS and DATA. Hence we have chosen not to model PCS.

Due to high propagation delay in underwater networks, PCS is not effective as in terrestrial radio networks. Commonly used action upon getting a PCS is to refrain from transmissions until PCS is inactive. However for networks with latency, such decisions can be counter productive. What is important is that when a packet arrives at a receiver, it is not receiving another packet. In networks with negligible latency, PCS is near simultaneous with such interfering receptions and hence the PCS logic is meaningful, where as in UW networks, PCS at a node intending to transmit has lower correlation with the intended recipient's state at the time that the potential packet will reach it.

D. Single long DATA instead of batches

In many prior analyses, single long data packets were used instead of a batch of short DATA packets. Many commercial modems can transmit longer DATA packets compared to control packets. Forward error correction (FEC) codes are used to make such DATA packets robust. Control-DATA collisions do not necessarily destroy the DATA packet as a whole, but introduce errors that are recoverable through FEC. The analysis model in this paper can be adapted by setting $B = 1$ and choosing appropriate k_D and L_D.

The following is a brief qualitiative comparison between batch of short DATA and single long DATA packets. Using short DATA packets in a batch gives a smooth degradation in performance in terms of losses due to control-DATA packet collisions, since at least some of the packets in the batch may be successfully delivered. If single long DATA packets are used instead of a batch of shorter ones, beyond a certain error threshold the entire large packet will be lost and the loss behavior will be abrupt. On the other hand, longer DATA could allow FEC coding to be spread across a larger packet and therefore be more robust to errors. However, this involves higher processing power also as FEC codec complexity in general is an increasing function of packet duration.

Long single DATA also cannot allow features such as DATA-VCS. Packet train based DATA-VCS is a technique used in terrestrial radio wireless. DATA-VCS helps in achieving VCS even if both RTS and CTS from neighbors are missed, by listening in to any of the DATA packets that follow. If there are many short DATA packets, the probability of receiving some of them at least is high.

It may be good to have "synchronization portions" in such long DATA packets, to avoid the wastage of losing the packet as a whole. For example, the equivalent single long packet for a batch size $B = 50$ with $L_D = 0.5$, is a 25second packet and it will not be prudent to lose it fully, just because the detection preamble was missed (probability P_d). In other words, it is good to keep a batch model even for longer contiguous data transmission, and keep L_D not too large. We shall not attempt a study on optimum L_D in this paper.

E. Adaptive batch size

As seen in Section V-C, optimum batch size reduces with traffic rate. Below a certain traffic rate, the optimum batch size reduces to 1. A simpler DATA-ACK protocol (with no RTS-CTS) may be a better choice for such low loads, since it is inefficient to have the RTS-CTS overhead just to send one or very few DATA packets. Thus we envisage an adaptive version of the proposed protocol, which can vary the batch size (as specified through RTS and CTS) to suit the traffic, and also switch to DATA-ACK mode (without RTS-CTS handshaking) for very low traffic. A detailed study of such an adaptive protocol shall be presented as part of a future paper.

F. Optimum number of ACKs

We can re-write (14) as

$$\begin{aligned} s_b &= \left(1 + (1-k)^i\right) V' + Y' + (i-1)Z \\ &= q^i V + Y + (i-1)Z \end{aligned} \qquad (32)$$

where $q < 1$, V', V, Y', Y, Z are factors used for convenience. This form for s_b clearly shows that for a certain $i >= 1$, there will be a minimum. The exact value depends on the parameters such as k, W, N, l, B, D etc. We found that for some of the parameter combinations used in the simulations, $i = 3$ gave the minimum (e.g., $k = 0.7, L = 0.5, D = 0.5$ $(l = 1, t_A = 2), N = 3, W = 9, B = 5$). In some cases $i = 2$ or $i = 4$ or other values may be better. It is thus possible to set optimal

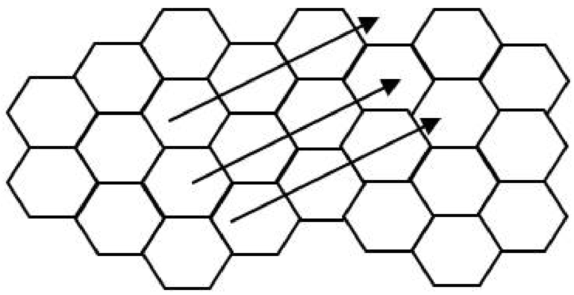

Fig. 18. Hexagonal cell model and sample traffic pattern

i according to network conditions, though we use a fixed i in this paper for simplicity. The optimum i can be derived as follows,

$$\begin{aligned} \frac{ds_b}{di} &= q^i \log_e(q)V + Z = 0 \\ i &= \left\lceil \frac{\log_e \frac{-Z}{\log_e(q)V}}{\log_e(q)} \right\rceil \end{aligned} \tag{33}$$

G. Multi-hop and hidden nodes

Although multi-hop networks are not the focus of this paper, we outline some thoughts on how the work in this paper can be extended to multi-hop networks in the future. The primary network UAN architecture considered in this paper is as described in [3] (reproduced in Fig. 17), where the acoustic networks are viewed as separate single collision domains connected to others via surface gateways. Although multi-hop is possible, the applications and implementations being pursued focused on single-hop networks. The analysis presented here is thus meant primarily for such isolated single collision domains and network architecture. Nevertheless, for large multi-hop acoustic networks, the results here could be used as an approximation in a N-node one-hop neighborhood. We shall give a suggestion on how this may be attempted for shallow water acoustic networks where 2D topology is reasonable. An in-depth analysis is out of scope for this paper and may be undertaken in the future.

A standard cellular hexagonal geometry may be used as shown in Fig. 18. Let us assume that the packet arrival rate at a node is Poisson with rate λ. We assume a traffic flow pattern as shown. Let H be the number of hops to the intended recipient of each transmission. We can consider two options, one where each node takes one hop at a time ($h = 1$) and another where a single hop is made across H nodes ($h = H$). In the traffic pattern shown, we can model the aggregate arrival rate at each node as $\lambda H/h$ considering routed traffic. To perform non-saturated queuing analysis we require $\lambda H < \mu$. The number of neighbors N in an effective collision domain is related to the number of hops h as shown below.

$$N = 6(1 + 2 + ... + h) = 3h(h + 1) \tag{34}$$

Assuming nearest neighbor connectivity (such connectivity may minimize power consumption [23]), this gives $h = 1$ and therefore $N = 6$. The aggregate traffic is λH. Thus, by using the effective number of nodes in the one-hop neighborhood and aggregate traffic ($N = 6$, traffic is λH in the example), we are able to get approximate estimates of MAC performance in multi-hop networks using the equations in this paper.

VII. Conclusion

We proposed enhancements for MACA based protocols for UANs such as monitoring of DATA packets during contention phase (to aid VCS) and Early-Multi-ACK. We derived a good analytical model that relates the parameters N, D, L, L_D, B, k, k_D, W, t_A to expected service time and throughput and specifically illustrated the impact of batch size (B) and detection and decoding probability (k, k_D) on throughput. Service time distribution was shown to be nearly exponential or more precisely Erlang-2, with an analytically known mean. Queuing analysis for Poisson arrivals was done and the effect of batch size B was shown. The throughput of the MACA protocol increases with batch size, but so does the waiting time and we have shown the existence of an optimum batch size. One of the important novel contributions is the formulation of a model that helps compute the total queuing delay for the retry based protocol. Optimum value of back-off counter W based on the number of nodes N has been derived. We implemented the protocols in acoustic modems, and medium range field trials corroborate the simulations and analysis well. System analysis is now possible without needing to resort to extensive simulations.

We note that for very long latency regimes, the slotted version of the protocol could be less effective, and the analysis results less accurate. In future, freezing back-off variation may be investigated for highly dense networks [4]. We may also analyze other variations to MACA or similar protocols through suitable modifications to the models used here. The relation between the packet sending procedure and the Erlang distribution is another problem for further investigation. Analysis of other traffic models such as bursty traffic, offers another interesting problem.

For large ad hoc UANs, where TDMA time frame assignment becomes cumbersome and its efficiency drops, MACA based protocols with packet trains and suitable variations such as Early-Multi-ACK is an excellent choice. Accurate analytical models of the protocol greatly helps in understanding its relationship to environmental and system parameters. Knowing the system performance before-hand can help to pre-determine possible throughput and waiting time so as to control data generation rate and plan realistic communication strategies for ad hoc UANs.

Appendix

A. The performance of the standard ACK model

Since there are no Early-Multi-ACKs, t_{B_VCS} in (11) is t_B (10), with $i = 1$. Instead of the Markov chain in Fig. 2 leading to (14), we can use Fig. 19 to compute batch service time as

$$s_b = \frac{1}{k}(s_{CTS} + t_B) = \frac{1}{k}(\gamma + Nt_B) \tag{35}$$

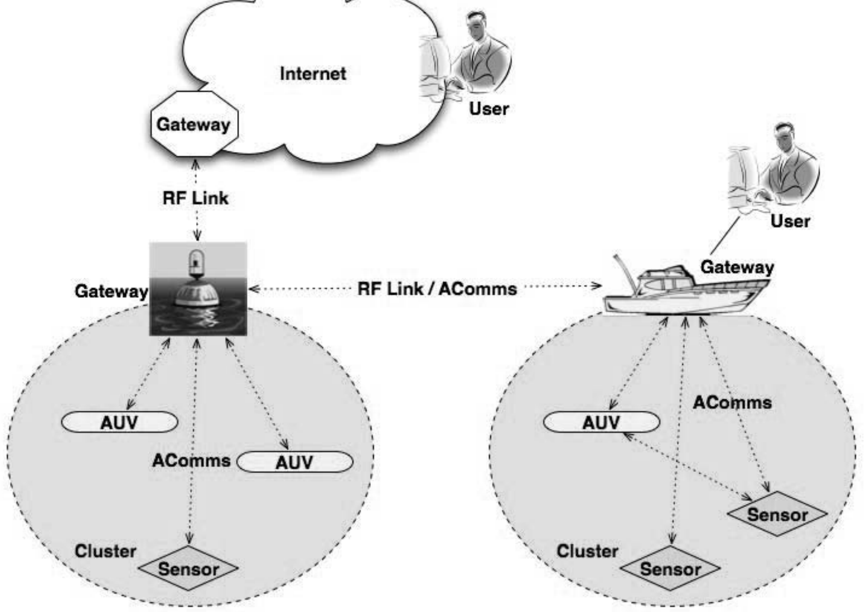

Fig. 17. UAN Architecture, adapted from [3]

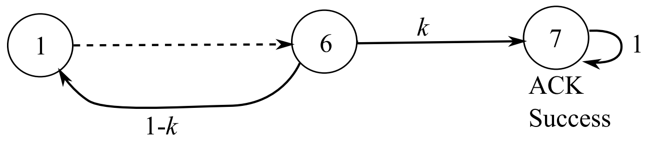

Fig. 19. Markov chain for Standard ACK model

B. Distribution analysis Markov chain

Here we show the distribution analysis matrix for the example described in Section IV. The state ordering is different here compared to (8) as detailed below. Each column represents states s_i. $\mathbf{M}_{i,j}$ is the transition probability from state s_i to state s_j. States s2 and s3 are the dummy states that represent the time delay of 2s corresponding to delay t_A (for given parameters) in state 3 in Fig. 1. States s_4, s_5, s_6, s_7, and s_8 are the dummy states that represent the time delay of 5s corresponding to delay t_B (from (10) for given parameters $B = 4, L = 0.5, l = 1, i = 3$) in both state 4 and state 5 (infact these states could be represented as one state) in Fig. 1. States s_9 and s_{10} are the dummy states that represent the time delay of 2s corresponding to delay t_A (for given parameters) in state2 in Fig. 1. States s_{11}, s_{12}, s_{13}, s_{14}, and s_{15} are the dummy states that represent the time delay of 5s corresponding to delay t_B (for given parameters) in state 6 in Fig. 1. The transition probabilities can then easily be seen by comparison with (8) and Fig. 1, with transition between dummy states having a probability of 1.

$$\mathbf{M} = \begin{bmatrix}
c & b & 0 & d & 0 & 0 & 0 & 0 & a & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
q & 0 & 0 & p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \tag{36}$$

For the retry scheme in Fig. 2, we can use $\mathbf{M}$ to generate the complete Markov matrix $\mathbf{M_R}$ (introduced in Section IV) as shown below, where $\eta = (1-k)^i$.

$$\mathbf{M_R} = \begin{bmatrix}
\begin{bmatrix} & & \\ & \mathbf{M} & \\ & & \end{bmatrix} & \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \vdots & \vdots \\ \eta & \dots & 0 \end{bmatrix} & \begin{bmatrix} 0 \\ \vdots \\ 1-\eta \end{bmatrix} \\
\begin{bmatrix} 0 & \dots & 0 \\ \vdots & \vdots & \vdots \\ 0 & \dots & 0 \end{bmatrix} & \begin{bmatrix} & & \\ & \mathbf{M} & \\ & & \end{bmatrix} & \begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix} \\
\begin{bmatrix} 0 & \dots & \dots & \dots & \dots & \dots & \dots & 1 \end{bmatrix} & &
\end{bmatrix} \tag{37}$$

REFERENCES

[1] M. Chitre, S. Shahabudeen, and M. Stojanovic, “Underwater acoustic communications and networking: Recent advances and future chal-

lenges," *Marine Technology Society Journal - "The State of Technology in 2008"*, vol. 42, no. 1, pp. 103–116, 2008.

[2] M. Stojanovic, "On the design of underwater acoustic cellular systems," in *IEEE Oceans'07*, Aberdeen, Scotland, 2007.

[3] S. Shahabudeen, M. Chitre, and M. Motani, *Underwater Acoustic Sensor Networks*. CRC Press, 2010, ch. Dynamic TDMA and MACA based Protocols for Distributed Topology Underwater Acoustic Networks.

[4] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[5] C. L. Fullmer and J. J. Garcia-Luna-Aceves, "Floor acquisition multiple access (FAMA) for packet-radio networks," in *SIGCOMM'95*, Cambridge, MA USA, 1995.

[6] B. Peleato and M. Stojanovic, "A MAC protocol for ad hoc underwater acoustic sensor networks," in *WUWNet'06*, 2006.

[7] C. Petrioli, R. Petroccia, and M. Stojanovic, "A comparative performance evaluation of MAC protocols for underwater sensor networks," in *IEEE Oceans'08*, 2008.

[8] X. Peng and J. H. Cui, "Exploring random access and handshaking techniques in large- scale underwater wireless acoustic sensor networks," in *IEEE Oceans'06*, 2006.

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

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

[11] C. G. Park, H. S. Jung, and D. H. Han, "Queueing analysis of ieee 802.11 MAC protocol in wireless lan," apr. 2006, pp. 139 – 139.

[12] P. Chatzimisios, A. C. Boucouvalas, and V. Vitsas, "Packet delay analysis of IEEE 802.11 MAC protocol," *Electronics Letters*, vol. 39, no. 18, pp. 1358–1359, 2003.

[13] C. Foh, M. Zukerman, and J. Tantra, "A markovian framework for performance evaluation of IEEE 802.11," *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1276–1265, 2007.

[14] N. Gupta and P. R. Kumar, "A performance analysis of the 802.11 wireless lan medium access control," *Communications in Information and Systems*, vol. 3, no. 4, pp. 279–304, 2004.

[15] P. Karn, "MACA - a new channel access method for packet radio," in *ARRL/CRRL Amateur Radio 9th computer Networking Conference*, 1990, pp. 134–140.

[16] S. Shahabudeen, M. Chitre, M. Motani, and A. Low, "Unified simulation and implementation software framework for underwater MAC protocol development," in *MTS/IEEE Oceans'09*, Biloxi, USA, Oct 26-30 2009.

[17] http://www.omnetpp.org/, "Omnet++, discrete event simulation system."

[18] http://arl.nus.edu.sg/twiki/bin/view/ARL/UNET#Modem, "Underwater acoustic communications, ARL."

[19] D. Gross and C. M. Harris, *Fundamentals of Queueing Theory*. Wiley, 1998.

[20] X. Guo, M. Frater, and M. Ryan, "Design of a propagation-delay-tolerant MAC protocol for underwater acoustic sensor networks," *IEEE Journal of Oceanic Engineering*, vol. 34, no. 2, pp. 170–180, 2009.

[21] http://sci.tech-archive.net/Archive/sci.stat.math/2010 03/msg00037.html, "MLE for Erlang distribution."

[22] M. L. Chaudhry and J. G. C. Templeton, *A first course in bulk queues*. John Wiley & Sons, 1984.

[23] A. P. Dolc and M. Stojanovic, "Optimizing the transmission range in an acoustic underwater network," in *OCEANS'07*, Vancouver, Canada, 2007.

Shiraz Shahabudeen received B.Eng degree in electrical engineering from the National University of Singapore (NUS) in 1998, and the M.S degree in telecommunication engineering from Melbourne University (Australia) in 2003. He worked in the telecommunication software industry from 1998 to 2002 and at the Infocomm Development Authority of Singapore (IDA) from 2003 to 2004 as a Wireless Technology Specialist. He joined ARL in 2004 and served as Research Fellow until 2010 where his research interests included underwater acoustic communications, networking and autonomous underwater vehicles. He currently works as a Senior Research Engineer at NeST Software, India.

Mehul Motani received the B.S. degree from Cooper Union, New York, NY, the M.S. degree from Syracuse University, Syracuse, NY, and the Ph.D. degree from Cornell University, Ithaca, NY, all in Electrical and Computer Engineering.

Dr. Motani is currently an Associate Professor in the Electrical and Computer Engineering Department at the National University of Singapore (NUS). He has held a Visiting Fellow appointment at Princeton University, Princeton, NJ. Previously, he was a Research Scientist at the Institute for Infocomm Research in Singapore for three years and a Systems Engineer at Lockheed Martin in Syracuse, NY for over four years. His research interests are in the area of wireless networks. Recently he has been working on research problems which sit at the boundary of information theory, networking, and communications, with applications to mobile computing, underwater communications, sustainable development and societal networks.

Dr. Motani has received the Intel Foundation Fellowship for his Ph.D. research, the NUS Faculty of Engineering Innovative Teaching Award, and placement on the NUS Faculty of Engineering Teaching Honours List. He has served on the organizing committees of ISIT, WiNC and ICCS, and the technical program committees of MobiCom, Infocom, ICNP, SECON, and several other conferences. He participates actively in IEEE and ACM and has served as the secretary of the IEEE Information Theory Society Board of Governors. He is currently an Associate Editor for the IEEE Transactions on Information Theory and an Editor for the IEEE Transactions on Communications.

Mandar Chitre received B.Eng. and M.Eng. degrees in electrical engineering from the National University of Singapore (NUS), Singapore, a M.Sc. degree in bioinformatics from the Nanyang Technological University (NTU), Singapore, and a Ph.D. degree from NUS. From 1997 to 1998, he worked with the ARL, NUS in Singapore. From 1998 to 2002, he headed the technology division of a regional telecommunications solutions company. In 2003, he rejoined ARL, initially as the Deputy Head (Research) and is now the Head of the laboratory. Dr. Chitre also holds a joint appointment with the Department of Electrical & Computer Engineering at NUS as an Assistant Professor. His current research interests are underwater communications, autonomous underwater vehicles, model-based signal processing and modeling of complex dynamic systems.

Dr. Chitre has served on the technical program committees of the IEEE OCEANS, WUWNet and DTA conferences and has served as reviewer for many international journals. He was the chairman of the student poster committee for IEEE OCEANS'06 in Singapore. He is currently the vice chairman of the IEEE OES (Singapore chapter) and the IEEE technology committee co-chair of underwater communication, navigation & positioning.