DESIGN, ANALYSIS, AND PERFORMANCE EVALUATION FOR HANDSHAKING BASED MAC PROTOCOLS IN UNDERWATER ACOUSTIC NETWORKS

NG HAI HENG

NATIONAL UNIVERSITY OF SINGAPORE

2012

DESIGN, ANALYSIS, AND PERFORMANCE EVALUATION FOR HANDSHAKING BASED MAC PROTOCOLS IN UNDERWATER ACOUSTIC NETWORKS

NG HAI HENG
(*B.Eng. (Hons), MMU*)

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2012

To my parents, and my beloved wife

Acknowledgements

First and foremost, I would like to express my sincerest gratitude to my supervisors, Assistant Professor Soh Wee-Seng and Associate Professor Mehul Motani, for having guided me patiently throughout the course of this work. Without their insightful suggestions, positive criticism, contributions and constant encouragement, this work would not have been possible. I feel honored to have an opportunity to work with them; they offer me a very enriching and enjoyable learning experience.

I would also like to thank Assistant Professor Mandar Chitre and Associate Professor Mohan Gurusamy, for their time and efforts to become the exam panel members of my Ph.D. qualifying examination. I really appreciate their valuable and constructive comments on my research. I am also indeed grateful to National University of Singapore for granting the four-year research scholarship that covers my monthly stipend, tuition fees, as well as conference expenses.

I am very thankful for my fellow members in the Communications and Networks Laboratory. My special thanks goes to Dr. Luo Tie for his useful comments and suggestions on my research, as well as many hours of thought-stimulating discussions. Many thanks to my friends and fellow lab members, Dr. Nitthita Chirdchoo, Dr. A.K.M. Mahtab Hossain, Dr. Hu Zhengqing, Dr. Ai Xin, Dr. Zeng Zeng, Dr. Zeng Linfang, Dr. Wang Yang, Hu Menglan, Yunye Jin, Chua Yu Han, Borhan Jalaeian, Neda Edalat, Ganesh Iyer, John Lau Kah Soon. Their friendship and support have made my Ph.D. experience both more educational and fun. Also, a big thank you to my laboratory technologist, Eric Poon Wai Choong and Goh Thiam Pheng, for their technical assistance in the lab.

In closing, I would like to express my heartfelt thanks to my parents and my two younger sisters. They have always provided unconditional support, love, and encouragement for me. Finally, a big thank you to my beloved wife, Sze Yin, for her patience, care, and love. Without my family, I could not have accomplished this journey.

Table of Contents

Abstract

Underwater wireless communication mainly relies on acoustic waves. Its unique characteristics like slow propagation speed and low bit rate-distance product present new challenges to Medium Access Control (MAC) protocol design. In this dissertation, we focus on the design, evaluation, and analysis of handshaking-based MAC protocols. By exploiting the acoustic channel's unique characteristics, we address the issues of: (i) how to adapt the original multiple access collision avoidance (MACA) protocol for use in multi-hop underwater acoustic (UWA) networks, (ii) how to improve channel utilization of handshaking-based MAC protocols, which in turn will offer both throughput and delay gains, (iii) how to accurately analyze the saturation throughput of slotted BiC-MAC (one of our proposed MACs) in single-hop networks, and (iv) how to better evaluate throughput performance of MAC protocols in static multi-hop wireless networks.

We first present a simple, adapted MACA MAC protocol, which can serve as a reference MAC for a better performance benchmarking in UWA networks. It is necessary because the evaluation against terrestrial handshaking-based MACs does not yield any meaningful insight, as they are not designed for high latency network. Our protocol has additional state transition rules to handle certain problematic scenarios that are likely to occur in multi-hop UWA networks. Furthermore, the packet forwarding strategy and backoff algorithm are modified as well.

Then, we propose a new approach to improve channel utilization. Here, a technique of bidirectional, concurrent data packet exchange is employed to improve the data transmission efficiency. To further amortize the high latency overhead, we also present a packet bursting idea, where a sender-receiver pair can exchange multiple rounds of bidirectional packet transmissions. We then design a single-channel, sender-initiated handshaking-based protocol called BiC-MAC, which does not require any clock synchronization. Our approach is more efficient than most conventional protocols, which often adopt a unidirectional packet transmission.

By exploiting the long propagation delay in a different way, we present another approach based on reverse opportunistic packet appending, to enhance channel utilization. An initiating sender can coordinate multiple first-hop neighbors to opportunistically transmit their appended data packets, with partial overlap in time. After the sender finishes transmitting its packets to its own receiver, it starts to receive the incoming appended data packets from different appenders, which arrive in a collision-free manner. Using this idea, a single-channel handshaking-based MAC called ROPA is proposed, where clock synchronization is also not needed. Unlike BiC-MAC, it does not impose rigid constraints on the packet size and inter-nodal distance; it complements BiC-MAC for a shorter range network.

Next, we propose an accurate analytical framework based on absorbing Markov chain to analyze the saturation throughput of slotted BiC-MAC in single-hop networks, under both error-free and error-prone channel conditions. As time slotting will lose its effects when inter-nodal propagation delay is much longer than a single control or data packet's duration, the analyzed results can serve as an approximation for the unslotted counterpart. We model the protocol behavior of a single tagged node, as it attempts to exchange its backlogged batch of data packets with its intended receiver, via bidirectional-concurrent transmission approach.

Finally, we revisit the use of throughput metrics in evaluating MAC protocols in static multi-hop wireless networks with negligible propagation delay. To complement existing single-hop and multi-hop throughput notions, we present a unified normalized throughput expression. Since current multi-hop metrics do not give much intuition on how close a MAC protocol's throughput is to the best achievable for a given network, we propose a new metric that benchmarks against the maximum achievable throughput. This proposed metric is also extended to evaluate three of our proposed MACs, in long propagation delay environment.

List of Tables

List of Figures

List of Abbreviations

ACK	Acknowledgement packet
AIDC	Appending Induced Data Collision
AODV	Adhoc On-Demand Distance Vector routing
ARQ	Automatic Repeat Request
AUV	Autonomous Underwater Vehicle
BEB	Binary Exponential Backoff
BIDC	Bidirectional Induced Data Collision
BILP	Binary Integer Linear Programming
CDMA	Code Division Multiple Access
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CTA	Clear-To-Append
CTS	Clear-To-Send
DSR	Dynamic Source Routing
FDMA	Frequency Division Multiple Access
MACA	Multiple Access Collision Avoidance
MAT	Maximum Achievable Throughput
NTF	Notification packet
MAC	Media Access Control
PER	Packet Error Rate
RTA	Request-To-Append
RTS	Request-To-Send
RTT	Round-Trip Time
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UWA	Underwater Acoustic

Chapter 1

Introduction

Unlike terrestrial wireless communication which uses radio waves, underwater communication mainly relies on acoustic waves [1, 2]. While terrestrial wireless networks have been studied extensively and well-established, researches on underwater acoustic networks have only recently begun, and still in infancy stage. Nonetheless, underwater acoustic networking is an important research area with tremendous practical potential; it could enable a diverse set of applications such as seismic monitoring, tsunami warning, mine reconnaissance, environmental monitoring, undersea explorations, distributed tactical surveillance, etc.

In underwater acoustic networks, we need to deal with the multiaccess problem, since the acoustic channel is shared across multiple distributed nodes. To this end, the use of an efficient Medium Access Control (MAC) protocol is of great importance, as it directly determines how effectively the competing communication nodes could access the shared acoustic channel. In the seven-layer Open Systems Interconnection (OSI) reference model, which developed as an international standard for data networks by the International Standards Organization (ISO), MAC protocol is part of the data link layer (layer 2 in OSI model), and it sits on top of the physical layer (layer 1) [3]. Since the MAC protocol directly controls a node's transceiver operation, it would have a huge impact on the network performance such as throughput, delay, energy consumption, etc. In this dissertation, we focus on the protocol design, performance evaluations and theoretical analysis on one popular class of MAC, called handshaking-based MAC protocol.

1.1 Background and Motivation

We now give some background information for the unique characteristics of underwater acoustic communication, and handshaking-based MAC protocols. The motivation behind the dissertation is also explained.

1.1.1 Underwater Acoustic Communication

While some of the underwater networking design approaches share some similarities with that of the terrestrial wireless networks, there are some fundamentally different challenges and research problems due to the use of acoustic communication. In general, both radio and optical communications are not practical in underwater environment. Radio waves suffer from strong attenuation in water, and thus have extremely limited propagation distance in the order of several meters (e.g., $1-8$ kbps at 122 kHz carrier for ranges up to $6-10$ m [4]). Although radio waves can propagate at long distances through conductive salty water, at extra low frequencies ($30-300$ Hz), it would be impractical due to large antenna requirement and high transmission power [1]. On the other hand, scattering and absorption are the major problems for optical waves, which limit its usage to very short-range communication. It has been reported that in very clear water, optical modems can achieve data rates up to several Mbps at ranges up to 100 m [5]. In [4, 6], optical communication is considered for low-cost, short-range links of around $1-2$ m, at standard IrDA rates such as 57.6 kbps. Hence, in order to allow a much longer communication range, acoustic waves appear to be a good practical choice [7].

There are two unique characteristics that arise from the acoustic communication, which significantly differ from the terrestrial wireless networks and should be carefully considered in the networking protocol design. First, underwater channel has a narrow and low bandwidth, that depends on both range and frequency; this results in low data rates. The acoustic bandwidth is severely limited due to absorption and the existing systems' range-rate product can hardly exceed 40 km-kbps [8]. A long-range system that operates over several tens of kilometers may have a bandwidth of only a few kilohertz, while a short-range system that operates over several tens of meters may have more than a hundred kilohertz of bandwidth [1]. Hence, unlike terrestrial networks, lower data rate in the order of

kbps is expected in underwater scenario. Second, the speed of sound in underwater is around 1500 m/s; the actual speed varies between 1433 and 1554 m/s, which depends on temperature, pressure and salinity. This is five orders of magnitude lower than radio waves' propagation speed of 3×10^8 m/s. In addition, the existing underwater node deployment is generally sparser than terrestrial networks (typically in the range of kilometers), due to the high cost of the nodes [1, 7]. Consequently, a transmitted packet in underwater often experiences extremely long propagation delay in the order of several seconds, before reaching its receiver (i.e., 0.67 s/km). This long delay characteristic adversely affects the network protocol's performance, especially in both throughput and delay. Many of the terrestrial MAC protocols, which are designed for high data rate and negligible propagation delay, perform inefficiently when applied blindly into underwater networks.

The acoustic signals suffer from transmission loss, multi-path and Doppler spread, in which these effects are more serious than terrestrial wireless counterpart [1]. The transmission loss can be attributed to two components, namely, the attenuation and geometric spreading. The attenuation loss is caused by signal absorption, in which the acoustic energy is converted into heat. It increases with frequency and distance. The geometric spreading is the dispersion of sound energy from the expansion of wavefronts. It is independent of frequency, but grows with distance. Multi-path propagation phenomenon is common in underwater channels, which results in inter-symbol interference (ISI). It is time-varying in nature due to surface waves and vehicle motions [7]; the severity of multi-path interference highly depends on the depth and the inter-nodal distance between a sender and its receiver. In a dynamic environment (e.g., moving platform like ships and scattering of the moving sea surface), the slow propagation speed of sound also yields a large Doppler spread, which causes interference among different frequency components of the acoustic signal. Moreover, acoustic communication has higher bit error rates compared to terrestrial wireless channel, as well as experiencing temporary losses of connectivity (i.e., shadow zones) due to frequency-dependent attenuation [1, 7].

To sum up, these channel impairments of transmission loss, multi-path interference and Doppler spread problems can be addressed via physical layer techniques; while the low data rates due to narrow bandwidth and long propaga-

tion delay, would have a major impact on the networking stack such as MAC layer and these characteristics should be accounted for in the protocol design.

1.1.2 Applicability of Different MAC Techniques

Generally, MAC protocols can be categorized into two major classes, namely, contention-free protocols and contention-based protocols. Contention-free MAC protocols include Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA) and Code Division Multiple Access (CDMA). The channel resources are deterministically separated in frequency, time and code domains, as such no packet collision is resulted. FDMA is rarely used, as it performs inefficiently due to the need of guard bands in the already limited bandwidth [9]. The limited band systems are also vulnerable to fading and multi-path [1]. TDMA can offer better performance [10]. However, its throughput is still very low due to the long guard time requirement. Furthermore, it demands a precise time synchronization, which is quite costly to achieve in underwater channels. CDMA is reported to perform better than TDMA and FDMA in certain scenarios [7, 11]. However, it demands a strict synchronization and power management mechanism; also, it is not clear how the near-far problem in underwater channel can be effectively addressed [7]. Finally, these contention-free protocols are inherently non-scalable [11], which is a concern for underwater deployment.

Unlike contention-free protocols, channel resources are not assigned a priori in the contention-based protocols. Example of contention-based protocols include Aloha [3], Carrier Sense Multiple Access (CSMA) [12] and handshaking-based MAC protocols [13–16]. These protocols offer benefits such as simplicity, flexibility and scalability; however, packet collisions could occur and MAC protocol requires a collision resolution algorithm. The Aloha has lower packet delay as it transmits directly whenever a packet arrives. But, it cannot maintain its throughput stability as offered load grows, due to the lack of packet collision avoidance mechanism [3]. To avoid excessive collision, CSMA performs carrier sensing by listening to the channel activity, before transmitting its packet. However, in multi-hop networks, CSMA performs poorly due to the prevalent of hidden node and exposed node problems [13, 14, 17]. A hidden node is one that is within the interfering range of the intended destination but out of the sensing range of the sender. Hence, carrier sensing at the initiating sender does not prevent packet collision at the

receiver node. In contrast, an exposed node is one that is within the sensing range of the sender but out of the interfering range of the destination. Exposed nodes can cause the available bandwidth to be under-utilized. Here, an initiating sender could potentially transmit without packet collision, albeit the channel is busy. More importantly, in long propagation delay, the carrier sensing mechanism in CSMA is ineffective in preventing packet collision [2]; even when a channel is sensed idle at a give node, it does not ensure that a packet is not already in transmission at a remote node.

Among the existing underwater MAC protocols, there is a strong focus on handshaking-based protocols, as they work well in multi-hop networks [1, 7, 11]. In fact, in the practical Seaweb project [9], they were shown to be more effective for underwater use compared to contention-free protocols and Aloha. In handshaking protocols, prior to the transmission of a long data packet, a series of small control packets will first be exchanged; this reduces the likelihood of data collision by reserving the floor around both sender and receiver nodes. A highly popular MAC from this family is called Multiple Access Collision Avoidance (MACA) [13], which uses a sender-initiated handshake. A sender and its intended receiver use a broadcasted Request-To-Send (RTS) and Clear-To-Send (CTS) packet, respectively, to reserve the floor. Any neighbor that overhears the control packets will defer its transmission for a specific amount of duration. MACA does not use carrier sensing; instead, it relies on packet sensing mechanism (also called virtual carrier sensing), in which the expected busy durations can be carried in the control packets so that an overhearing node is aware of channel activity [13].

In multi-hop underwater networks, MACA-based protocols can offer multi-fold benefits such as: (i) carrying of useful information in the control packets such as modulation parameter [9], (ii) alleviating the hidden and exposed node problem, (iii) reducing collision cost due to small control packet sizes, and (iv) allowing a simple, decentralized network operation, in which time synchronization is not needed. The handshaking-based method is even more useful, especially for MAC with packet train enhancement.

However, the original MACA still suffers from low throughput and large delay in underwater; specifically, it does not handle certain problematic scenarios that arise in long propagation delay. Furthermore, a large overhead is resulted due to the multi-way handshake, and only a single packet is exchanged for each

successful handshake. In general, any handshaking-based protocol design should also consider the narrow bandwidth and low data rate characteristics; thus, a single-channel MAC design is desired, and the control packet overhead must be minimized. Finally, node mobility due to underwater currents, must be catered for in the design.

1.2 Research Objectives

The research objectives of this dissertation are as follows:

1. We aim to adapt the original, terrestrial-based MACA protocol for use in multi-hop UWA networks. This is to be accomplished by modifying the operation rules of the original MACA to handle potential problematic scenarios, which only arise due to the long propagation delay. The adapted protocol will serve as a benchmarking protocol for more advanced underwater handshaking-based MAC protocols.

2. We aim to enhance channel utilization of handshaking-based MAC protocols, which in turn will offer performance gains in both throughput and delay. This is to be achieved by designing MAC protocols that not only seek to reduce communication overheads, but also improve data transmission efficiency in UWA networks. The packet exchange mechanism in our proposed protocols are meticulously designed to exploit the simultaneous transmission opportunity, offered by the slow propagation speed of sound in water.

3. We aim to analytically compute the normalized saturation throughput performance of a time slotted BiC-MAC protocol in single-hop networks (note that BiC-MAC is one of our proposed protocols that employs a bidirectional packet exchange approach, which will be explained later). To attain this, a detail analytical framework is proposed to model the protocol behavior of BiC-MAC, as a sender-receiver pair intends to exchange their data packets bidirectionally. We also study how the analytical results can be used to closely approximate the unslotted BiC-MAC's saturation throughput.

4. We aim to better compare and evaluate the throughput performance of MAC protocols in static multi-hop wireless networks, in which the evaluation will yield as much intuition as the single-hop throughput metric, with regard to

the performance relative to best achievable bit-rate. This is to be achieved by using a new throughput metric, that accounts for the maximum achievable throughput in a given multi-hop network topology.

1.3 Main Contributions

The following summarizes the main contributions from this dissertation:

1. The adaptation of the conventional MACA protocol (3-way RTS/CTS/DATA handshaking-based MAC) for multi-hop UWA networks; three key areas of improvement are identified: (i) state transition rules, (ii) packet forwarding strategy, and (iii) backoff algorithm, and modified accordingly so as to account for the long propagation delay characteristic in underwater networks. Via simulation, we have shown that the adapted MAC achieves a stable throughput, and improves throughput efficiency compared to the original MACA that applied blindly into underwater networks. Due to its protocol simplicity, the adapted protocol can be used as a more appropriate reference MAC for benchmarking of underwater handshaking-based MAC protocols.

2. The design of an asynchronous, sender-initiated handshaking-based MAC that utilizes a novel approach of bidirectional, concurrent data packet exchange, so as to improve data transmission efficiency. To further amortize excessive communication overheads caused by long propagation delay, a packet bursting idea is adopted, that allows a sender-receiver node pair to exchange multiple round of bidirectional packet transmissions. For more flexibility, a versatile framework is also conceived so that our MAC can operate in three possible bidirectional transmission modes. Unlike many existing protocols that only allow for unidirectional transmissions, our MAC is the first to use a comprehensive bidirectional, concurrent transmission MAC framework for exchanging data packets in UWA networks. Via simulation and comparison with existing MAC protocols, our protocol has shown the value of adopting a bidirectional-concurrent transmission approach in high latency networks, where it greatly improves both throughput and delay performance.

3. The design of an asynchronous, sender-initiated handshaking-based MAC that uses another novel approach – reverse opportunistic packet appending,

to exploit the simultaneous transmission opportunity in UWA networks. In each handshake, an initiating sender can schedule its first-hop neighbors to transmit their appended packets with partial overlap in time in such a way that these packet trains will arrive at the sender in a collision-free manner, soon after it finishes transmitting its own packet train to its intended receiver. This not only helps to reduce the proportion of time spent on control signaling, but also achieves a better channel utilization. Our method is in contrast to the conventional approach, which requires each of those neighbors to initiate a separate handshake that incurs its own overheads.

4. The development of a simple analytical framework based on absorbing Markov chain, for computing normalized saturation throughput of slotted BiC-MAC in single-hop networks, under both error-free and error-prone channel models. Our model captures the protocol behavior from a single tagged node's perspective, as it attempts to bidirectionally exchange its backlogged batch of data packets with its intended receiver. In order to obtain its average batch service time (used for throughput computation), the state transition probabilities and expected time durations that a node spent in each state, have been derived. From our validation against simulated slotted BiC-MAC in small and large networks, we have shown that our model can give very accurate saturation throughput results. In addition, a throughput approximation approach that utilizes the information of actual inter-nodal delays in the analytical expression, is also proposed. From our evaluation, we found that it can closely approximate the saturation throughput of unslotted BiC-MAC, in which nodes are randomly deployed in a single-hop square area.

5. The proposal of a unified normalized throughput expression, that allows the existing normalized throughput metrics of both single and multi-hops to be expressed in a general formula. Moreover, a new multi-hop throughput metric is also presented, that benchmarks against the maximum achievable throughput in a given static multi-hop wireless networks with negligible propagation delay. We have demonstrated its use to evaluate the conventional Aloha and CSMA/CA MAC protocols in both string and square grid topologies. Unlike the existing throughput metrics, our metric can offer more intuition on a MAC protocol's relative performance to the best achievable.

The metric is also extended for evaluating our proposed MAC protocols in these two topologies, under the presence of long propagation delay.

1.4 Organization of the Thesis

The remaining of this dissertation is organized as follows. Chapter 2 presents literature survey focusing on the representative UWA MAC protocols, as well as related works on throughput analysis of MAC protocols. Chapter 3 introduces a simple handshaking-based MAC protocol, in which its protocol's operation rules are adapted from the original MACA MAC protocol for the use in multi-hop UWA networks. The adapted protocol is intended to serve as a more appropriate benchmarking MAC. Chapter 4 presents the design and performance evaluation of a MAC protocol that utilizes a novel approach of bidirectional, concurrent data packet exchange in UWA networks; unlike most existing protocols that adopt unidirectional data transmission, our protocol achieves a better channel utilization and offers significant performance gains in terms of both throughput and delay. Chapter 5 describes another sender-initiated handshaking-based MAC protocol that aims to offer high channel utilization; here, a novel approach based on reverse opportunistic packet appending is proposed. Chapter 6 provides an accurate analytical framework based on absorbing Markov chain to compute the normalized saturation throughput for slotted BiC-MAC, in single-hop networks. We also demonstrate how the analytical results of slotted variant can serve as a reasonably well approximation for the throughput performance of an unslotted BiC-MAC counterpart. For a better throughput comparison across different MAC protocols, Chapter 7 presents a new throughput metric, that benchmarks against the maximum achievable throughput in a static multi-hop wireless networks with negligible propagation delay; this gives more insight with regard to the protocol's performance relative to the best achievable. We also utilize this metric to evaluate our proposed MAC protocols in a long propagation delay environment. Finally, Chapter 8 concludes and reviews our research contributions, as well as outlines potential directions for future research.

Chapter 2

Literature Survey

We review related works on: (i) the proposed MAC protocols for UWA networks, and (ii) throughput analysis of MAC protocols. For both sections, we shall emphasize on the research efforts for underwater handshaking-based MAC protocols.

2.1 Underwater MAC Protocols

We first focus on the existing underwater handshaking-based MAC protocols; then, we shall briefly describe some proposals for non-handshaking protocols.

Handshaking-based MAC protocols can be divided into two categories: sender-initiated and receiver-initiated. For the former category, some proposed protocols only allow a sender to transmit a single data packet unidirectionally for every successful handshake. Early work such as the Seaweb 2000 experiment [9] relies on 3-way RTS/CTS/DATA handshake. This RTS/CTS exchange is also exploited as a channel probing mechanism to determine the node-to-node range, impulse response, signal-to-noise ratio (SNR), and optimal transmit power. Similarly, Doukkali and Nuaymi [18] also exploit the handshake as channel probing mechanism by incorporating transmission power control. However, the probed channel condition may not be accurate as the probe packets experience long propagation delay. To overcome unreliable channel condition, Sozer *et al.* [10] propose a 3-way RTS/CTS/DATA handshake MAC protocol with error detection via Stop-and- Wait Automatic Repeat Request (ARQ). The authors suggest that a busy destination node can broadcast a WAIT control packet to the initiating sender, so as to alleviate the repetitive transmission request problem. In [19], Molins and Stojanovic propose the Slotted-FAMA that utilizes 4-way

RTS/CTS/DATA/ACK handshake; it employs a time-slotting mechanism so that all packets are transmitted at the beginning of time-slots. Although Slotted-FAMA achieves guaranteed data collision avoidance, the long slot length requirement leads to very low throughput performance. Furthermore, it also demands precise time synchronization. Guo *et al.* [20] introduce a 3-way handshake protocol called Adaptive Propagation-delay-tolerant Collision Avoidance Protocol (APCAP), that allows a sender to take actions for other packets in its buffer while waiting for a CTS packet to return. This allows a sender to have multiple reservations concurrently, and thus improve channel utilization. However, its packet delay can potentially be very large, since both intended receiver and sender of APCAP may deliberately delay the responses of CTS packets, and data frames, respectively. The authors also do not describe its packet scheduling mechanism in detail. To improve throughput efficiency, Peleato and Stojanovic [21] propose another 3-way handshake protocol called Distance Aware Collision Avoidance Protocol (DACAP). It allows a sender to use different handshake lengths (which determined from inter-nodal separation distance) for different receivers, so as to minimize the average handshake duration. DACAP also has a collision avoidance mechanism, in which an intended receiver can send a WARNING control packet to the sender if it deduces that a packet collision might occur. Upon receiving the WARNING packet, the sender aborts its data transmission. In [22], the authors incorporate discrete power control to DACAP protocol. They show that the mechanism is necessary for enabling a scalable, large coverage multi-hop communications in the narrow bandwidth underwater networks. Finally, Kebkal *et al.* [23] propose a MAC protocol with RTS/CTS/DATA1/ACK/DATA2 handshake for point-to-point connection. Each complete data packet is partitioned into a larger DATA1, and a smaller DATA2 portion. A pair of nodes shall take turns to send their DATA1 portions unidirectionally, and then exchange their DATA2 portions concurrently. Generally, these aforementioned MAC protocols often have poor throughput and delay performance in underwater, since they have a large control packet overhead (exaggerated by long propagation delay), as only a single data packet is transmitted unidirectionally from a single source node.

To further improve channel utilization, some handshaking-based protocols allow an initiating sender to transmit multiple data packets back-to-back (i.e., packet train) unidirectionally, for each successful handshake. In [19], the authors

describe how packet train mechanism can be employed in Slotted-FAMA to increase its throughput efficiency. However, they do not include any simulation results. Shahabudeen *et al.* [24] propose the MACA with Multiple Channels and Positioning information (MACA-MCP) protocol for autonomous underwater vehicle (AUV) networks. The authors assume that each mobile AUV simultaneously uses three acoustic modems that operate at different frequency bands and different ranges. Their protocol employs a 4-way RTS/CTS/DATA/ACK handshake with packet train enhancement. Nonetheless, it could be too costly for practical implementation. In [25], Chirdchoo *et al.* propose a 3-way handshake protocol called MACA with packet train to Multiple Neighbors (MACA-MN); here, an initiating sender can send its packet train to multiple neighbors in a single handshake. However, due to the long duration of each handshake, the average waiting time can be very long before a node gains control of the channel to transmit. Xie and Cui [26] propose an energy-efficient, Reservation-based MAC (R-MAC) protocol, which operates in a periodic active/sleep cycle. R-MAC schedules the transmission of control packets and data packets at both sender and receiver to avoid data collisions. They adopt the packet train approach in their data transmission phase. But, their protocol is mainly designed to achieve energy-efficiency and fairness, rather than offering a high throughput.

Finally, we review the receiver-initiated MAC category. So far, we only found one MAC instance. In [27], Chirdchoo *et al.* propose the Receiver-Initiated Packet Train (RIPT) protocol that relies on a 4-way RTR/SIZE/ORDER/DATA handshake. It uses receiver-initiated reservations to schedule packets from multiple neighbors to arrive at the receiver node, in a packet train manner. However, the receiver-initiated approach often demands a complex traffic prediction algorithm. In contrast, the sender-initiated approach is more intuitive and suitable for common traffic pattern in generalized networks. Also, its throughput is still low, due to inefficient handshake design. As an example, a large gap is always present in RIPT just before the arrival of the first packet train at the receiver node.

There are some works carried out to adapt or evaluate the existing terrestrial MAC protocols for underwater. Shahabudeen and Chitre [28] study the performance of both Aloha and handshaking-based protocols, along with orthogonal and non-orthogonal physical layer models. In [29], Stojanovic evaluates three variants of Stop-and-Wait automatic repeat request (ARQ) protocols; the author

demonstrates that the throughput efficiency can be significantly increased when a group of packets (i.e., packet train) is transmitted and acknowledged selectively. Furthermore, optimal throughput can be achieved when the packet size is carefully chosen, which is a function of range, data rate and channel error probability. Xie and Cui [30] perform an analytical study on Aloha and handshaking-based protocols in single-hop networks. By using two illustrative examples, the authors contend that the normal 3-way RTS/CTS/DATA handshake should be modified for the long propagation delay scenario. Nonetheless, they do not provide any protocol operation rules in detail. Foo *et al.* [31] propose some simple adaptations to a scheme based upon the Ad-hoc On Demand Distance Vector (AODV) routing protocol coupled with original MACAW [14] protocol. Since it is more likely that two nodes may transmit to each other in high latency network, they suggest to use priority rules for resolving the conflict.

Lastly, we survey some other representative underwater MACs, that do not adopt the handshaking technique. In [32], the authors propose UWAN-MAC, which aims to achieve energy-efficiency by employing periodic active/sleep cycle, as well as trying to minimize packet collisions. Each node can schedule its transmission time for its next packet, and broadcast this schedule by carrying the information in its current data packet. Upon overhearing this, its neighbors know when to wake-up for receiving the next packet. This protocol does not demand strict synchronization, as a node only advertises the time interval between the wake-up event, instead of absolute wake-up time. However, it has a low throughput, since a small duty cycle is maintained for minimizing packet collisions.

In [33], Chirdchoo *et al.* propose Aloha-based protocols, namely, the Aloha with Collision Avoidance (Aloha-CA) and Aloha with Advance Notification (Aloha-AN), for use in single-hop networks. In Aloha-CA, a node pays close attention to each overheard packet, so as to get the information the sender and its intended receiver's node IDs. By using this information, along with the knowledge of inter-nodal delays of all node pairs, a node can avoid collision by computing the expected busy durations. To further improve throughput efficiency, an initiating sender in Aloha-AN, will first transmit a notification packet, prior to a data packet transmission. This additional packet transmission delay allows a sender to collect more useful information about its neighbors' activities, which helps in alleviating collisions. Nonetheless, their throughput is still quite low, as compared to our

handshaking-based MACs in multi-hop networks.

Kredo *et al.* [34] propose a scheduled, collision-free TDMA-based MAC called Staggered TDMA Underwater MAC Protocol (STUMP). To improve channel utilization, it uses propagation delay estimates to schedule overlapping transmissions, as such no collision is resulted at an intended receiver. Specifically, STUMP uses node position diversity to overlap communications, where an initiating sender divides the area around itself into concentric logical rings, so as to perform finer packet scheduling. To find the optimal schedule, the authors present both centralized and distributed scheduling algorithms, where a linear programming problem needs to be solved. While the centralized scheduling could be vulnerable to a single-point failure, the distributed algorithms require a network-wide knowledge for solving the scheduling problem.

2.2 Throughput Analysis of MAC Protocols

We now examine the existing works on throughput analysis of MAC protocols in both terrestrial wireless and underwater networks. We shall focus our attention on handshaking-based MAC protocols.

2.2.1 Throughput Analysis of Terrestrial MAC Protocols

In terrestrial wireless networks, the RTS/CTS handshaking-based MAC technique has gained remarkable success and popularity; we now examine some representative works in analyzing its throughput performance.

In [15], the authors propose the floor acquisition multiple access (FAMA) protocol, which uses both non-persistent carrier sensing and 3-way RTS/CTS dialogue to reserve the "floor" around the sender and its intended receiver, before data packet can be successfully transmitted. Two conditions are given to ensure a data collision-free transmission: (i) the RTS duration should be greater than the maximum propagation delay, and (ii) the CTS duration should be two times greater than the maximum propagation delay plus the transceiver's transmit-receive turnaround time. While these conditions are easily satisfied in a network with negligible delay, it is inefficient to design a large control packet size in underwater networks. For their single-hop throughput analysis, they have derived closed-form formulas for both FAMA and MACA (it uses packet sensing, instead

of carrier sensing) protocols. They use a renewal theory approach similar to the Kleinrock and Tobagi's work in [12]; the average useful period, busy period and idle period, in each data transmission cycle are derived. In [17], the authors propose and analyze another approach to overcome the hidden and exposed node problems, which is based on the use of busy tones. The MAC is called Dual Busy Tone Multiple Access (DBTMA). It no longer needs the CTS packet; instead, its handshake relies on the RTS, along with two narrow-bandwidth, out-of-band busy tones. Their throughput analysis is quite similar to the FAMA's method. Unlike our analysis, both protocol models in the above analyses are much simpler, in which they assume an infinite node population and no backoff algorithm is considered. Also, they only consider an error-free channel model.

The IEEE 802.11 [16] Wireless Local Area Networks standard has received considerable research efforts. Its MAC layer, called Distributed Coordination Function (DCF), is based on the CSMA/CA with discrete time-slotted binary exponential backoff (BEB) algorithm. DCF specifies two mechanisms of transmitting a single data packet: (i) 2-way DATA/ACK handshake, and (ii) 4-way RTS/CTS/DATA/ACK handshake. Unlike the classical BEB, the DCF's backoff algorithm has a unique "freezing" feature. Specifically, the backoff counter is decremented as long as a backlogged node senses its medium as idle; the timer is frozen when the medium is sensed busy. After a busy period, a node can resume its decrementing of backoff counter when the medium is sensed idle for more than a short duration (known as distributed interframe space period). In his seminal work, Bianchi [35] analyzes the saturation throughput of DCF in single-hop networks, under an error-free channel condition. The author develops a two dimensional, regular Markov chain to model the backoff process of a single node. From the Markov model, the stationary probability that a node transmits a packet in a random slot, can be obtained. Using this transmission probability, the saturation throughput can be found by analyzing the events that can occur within a random slot (i.e., idle, collision, and successful transmission events). Since then, many works have extended the analysis to consider other protocol features from DCF's enhancement. For example, in [36], the authors extend the model to account for frame retransmission limit. In [37], the authors analyze the saturation throughput for IEEE 802.11e standard, which introduces the quality of service (QoS) support into the protocol. While the above works only consider

ideal channel conditions, some works also extend the Markov analysis to account for error-prone channel [38–40]. Recently, some studies also modify the Markov analysis, so as to relax the saturated traffic assumption [41–43]. In short, these aforementioned analytical frameworks are not applicable for our BiC-MAC analysis, mainly due to the fundamentally different protocol operation, as well as they do not take into account the long propagation delay characteristic.

2.2.2 Throughput Analysis of Underwater MAC Protocols

There are currently limited works on the throughput analysis of underwater MAC protocols; we shall focus on the analysis for random access MAC, such as the Aloha-based and handshaking-based MAC protocols.

In [44], Vieira *et al.* analyze the throughput performance of pure Aloha and slotted Aloha in long propagation delay environment, and show that the throughput of slotted Aloha would degrade to that of the pure Aloha. Here, the slot length is defined as the transmission time of single packet. Although a node sends its packet at the beginning of slot, it is highly unlikely that the packet will arrive at the starting of a time slot, at the receiving nodes. Hence, the slotting mechanism is no longer effective to contain the packet collision to occur within a slot boundary. In [45], Xiao *et al.* extend the Aloha-based analysis into multi-hop networks. The authors study the channel utilization of both Aloha and p-persistent Aloha protocols in multi-hop string network, with a single gateway node as final destination. To regain the benefit of time slotting, Syed *et al.* [46] consider the use of guard bands in each time slot; the slot length is defined as $T+\beta\cdot\tau_{\max}$, where T is the packet duration, $\tau_{\max}$ is the maximum inter-nodal delay, and $0 \leq \beta \leq 1$. However, they only demonstrate the throughput improvement via simulation, but not from theoretical analysis. Subsequently, in [47], Ahn *et al.* refer to this Aloha variant as Propagation Delay Tolerant Aloha (PDT-Aloha), and analytically study its throughput performance in single-hop networks (which has a single receiver and multiple transmitter nodes). With proper parameter settings, the authors show that the throughput of PDT-Aloha is $17-100\%$ better than that of the conventional slotted Aloha. These aforementioned analytical approaches are not applicable in our BiC-MAC analysis, mainly due to the significant differences in the operation rules between the Aloha-based and handshaking-based protocols.

We now review the analyses on handshaking-based MAC protocols. Earlier work in [19], the authors propose and analyze the throughput per-node of Slotted-FAMA, which relies on a 4-way RTS/CTS/DATA/ACK handshake. Similar to our study, the slot length is defined as a single control packet's transmission time plus maximum inter-nodal propagation delay. For analysis simplicity, a special network topology is used that leads to every adjacent node pair is separated by the same inter-nodal distance. To find the throughput per-node, the authors seek for the average useful period, busy period and idle period, in each transmission cycle. However, they do not consider the use of backoff mechanism prior to an RTS attempt. Instead, the data packet arrival process is assumed to follow a Poisson distribution, and a node shall broadcast its RTS at the beginning of the next time slot. While the authors provide a closed-form throughput expression, they do not validate their analytical result against the simulation performance; thus, it is not clear how accurate the analytical model is. Xie *et al.* [30] present a simple, single-hop throughput analysis of a conventional, unslotted 3-way RTS/CTS/DATA handshake MAC. In their setup, there is only a single receiver node and each transmitter is located at a fixed, identical distance to the receiver. The analysis also does not account for any backoff mechanism. Both the above analyses consider a much simpler protocol model. For example, an initiating sender only transmits a single data packet unidirectionally, for each RTS-CTS handshake. They also do not study the protocol's saturation throughput performance.

Some recent works [48, 49] focus on analyzing the saturation throughput of handshaking-based protocols in single-hop network. In [48], Aldawibi *et al.* adapt the terrestrial IEEE 802.11 Distributed Coordination Function (DCF) MAC [16] with minimal changes. The authors suggest that the Inter Frame Space (IFS) interval of IEEE 802.11 MAC should vary according to the inter-nodal delay of a transmitter-receiver pair. They directly use the Bianchi's [35] throughput formula to analyze the saturation throughput. While they show via simulation that the adapted MAC achieves a better throughput, it is not clear how good the analytical model is, since they do not verify their analytical results against simulation.

In [49], the authors study the saturation throughput performance of a time slotted MACA protocol with packet train mechanism (i.e., a handshake of RTS/CTS/DATA packet train/ACK), in single-hop networks. A novel ARQ enhancement called Early-ACK, is presented that allows a node to reply with

ACK instead of CTS packet, when RTS is repeated for the same packet train. They model the unidirectional data packet transmission process as an absorbing Markov chain, and derive a closed-form expression for the average batch service time (which in turn used to compute throughput). While they account for the error-prone channel model, they assume that the probability of successful packet reception, is same for both control and data packets; this may be oversimplified as control packets are usually encoded with stronger FEC codes due to they are crucial to setup a handshake. Moreover, for their slot definition, they assume that the maximum inter-nodal delay is much smaller than the transmission time of a single control or data packet. This condition does not allow for bidirectional packet transmission, which is adopted in our protocol. Although the authors have shown that the analysis approach can give a reasonably close approximation via simulation and sea trials, they limit their evaluations to a small topology of 4-node.

In [50], Zhou *et al.* analyze both multi-channel Aloha and RTS/CTS-based MAC protocols for throughput and energy consumption, in single-hop underwater networks. The channel resources are divided into m data channels, and a single control channel. Prior to the transmission of a single data packet, the initiating sender will send a control packet in the control channel to inform its intended receiver of the selected data channel. For analytical simplicity, the inter-nodal delay between any node pair is also assumed to be the same. In their RTS/CTS-based analysis, the authors first analyze the behavior of the control channel to obtain the effective traffic to data channels. Next, they analyze the packet collision behavior of the data channels, and develop two Markov models to give throughput upper bound and lower bound. Numerical methods are used to solve for the bounds. Their results show that the RTS/CTS protocol outperforms the Aloha protocol in most of their experiments, while the Aloha can achieve stable performance with varying propagation delays. In their model, however, a node does not employ backoff algorithm. They do not focus on saturation performance, and their model is for a unidirectional MAC with single packet exchange.

In [51], the authors study their proposed contention-based MAC, namely, the T-Lohi, in single-hop networks. It is designed for energy-efficiency and required a special hardware of energy tone detector, so that the primary transceiver that consumes more energy, is switch off most of the time when data is not transmitted/ received. Nodes contend using a short tone to reserve data transmission. After

sending a short tone, a node listens for the remaining duration of the contention round to decide if its reservation is successful. The node could transmit its data, if only a single node intends to transmit in the round. For the reservation period, there will potentially be multiple rounds of contentions, if more than one node wish to access the channel. The authors do not derive the protocol's throughput; instead, they model the contention process using Markov chain to seek for the average reservation duration, which is obtained by numerically solving the model.

While [15] has derived a closed-form throughput expression for MACA protocol in terrestrial wireless networks, the equations only hold when the duration of RTS and CTS, is at least twice as long as the maximum inter-nodal propagation delay. Matsuno *et al.* [52] analyze the throughput of MACA in single-hop networks, when the control packets' lengths are much shorter than the maximum propagation delay (i.e., typical scenario for underwater networks). The authors derive closed-form expressions for both throughput upper and lower bounds. They have assumed infinite number of nodes and packet arrival process follows the Poisson distribution. Moreover, the inter-nodal delay between any node pair is assumed to be same as the maximum delay. While they offer accurate closed-form expressions, the analysis approach is quite cumbersome, as it needs to consider a large number of potential data collision scenarios for deriving their respective busy periods. In addition, the protocol model also does not account for backoff algorithm.

Chapter 3

A Reference MAC Protocol for UWA Networks

3.1 Introduction

As described in Chapter 1, underwater communication is fundamentally different from the terrestrial counterpart mainly due to the use of acoustic waves; the acoustic communication can be characterized by unique properties such as slow propagation speed of sound in underwater, as well as low data rates in a limited bandwidth channel. As a result, terrestrial MAC protocols perform inefficiently when deployed directly in an underwater environment, since they are designed for networks with negligible propagation delay and high data rates.

In this chapter, we examine how a highly popular asynchronous handshaking-based MAC protocol called Multiple Access Collision Avoidance (MACA) [13] can be adapted for use in multi-hop underwater networks. While the original MACA is widely adopted as a reference MAC when evaluating more advanced terrestrial MAC protocols, it does not yield any meaningful insight, because the MACA protocol was not designed for high latency networks. Specifically, there are some problematic scenarios that may show up in MACA in such environments, which have not been addressed previously. Therefore, there is a need to modify the original MACA to accommodate such scenarios, before it can be used as a meaningful benchmark protocol. For our proposed protocol adaptations, we have identified three areas of modification, namely, the state transition rules, the packet forwarding strategy, and the backoff algorithm. Each of these areas will be carefully adapted by accounting for the long propagation delay in underwater

networks. The resulting MAC protocol is that we call MACA for Underwater (MACA-U). We will adopt MACA-U as a reference protocol to benchmark the performance of our proposed MAC protocols in Chapter 4 and Chapter 5.

The remainder of this chapter is organized as follows. In Section 3.2, the original MACA protocol is briefly described. In Section 3.3, we describe how the MACA protocol can be adapted for underwater multi-hop acoustic networks. Then, in Section 3.4, the simulation results for MACA-U are presented and discussed. Finally, we present in Section 3.5 the conclusion drawn.

3.2 Original MACA Overview

In MACA, a source node that has packet to send will contend for floor reservation by sending a Request-To-Send (RTS) control packet to the destination node. Upon receiving the RTS, the destination node immediately replies a Clear-To-Send (CTS) control packet back to the source node. MACA adopts the packet sensing mechanism, in which the proposed data transmission's length is embedded in the control packet. After receiving the CTS, the source node immediately sends data to the destination node. Any neighboring node that overhears a control packet that is intended for another node (xRTS or xCTS) will defer its transmission, and transit to QUIET state. The neighboring nodes remain in QUIET state until the corresponding CTS or data packet transmission would have finished. Therefore, data collision is minimized through the transmission deferment. In the event of CTS failure, which could either be due to CTS packet corruption or the destination node is busy, the source node shall schedule a packet retransmission using Binary Exponential Backoff (BEB) [16] algorithm. As mentioned in the previous section, there is a need to adapt the original MACA to accommodate some problematic scenarios in multi-hop underwater networks.

3.3 Proposed MACA Adaptation for Multi-hop UWA Networks

In this section, we introduce MACA-U and its associated adaptations for underwater networks. MACA-U has five distinct states, namely, IDLE, CONTEND (CTD), Wait-For-CTS (WFCTS), Wait-For-DATA (WFDATA) and QUIET. From

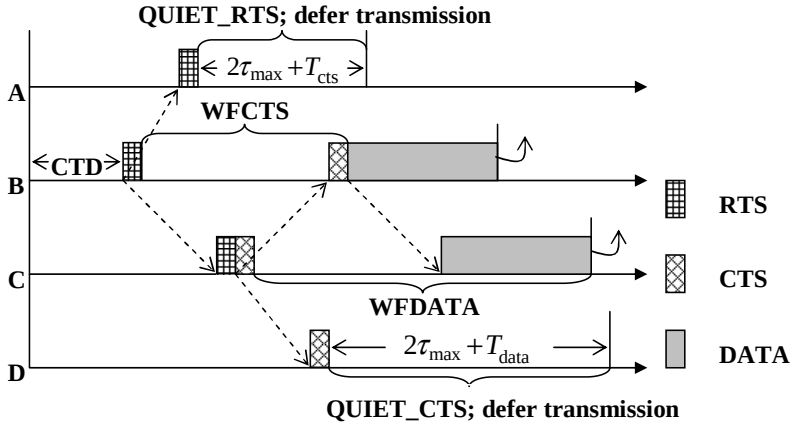


Figure 3.1: Timing diagram for MACA-U. While MACA-U also relies on 3-way RTS/CTS/DATA handshake, it adapts state transition rules to cater for long propagation delay, in which different actions are taken in the WFCTS, WFDATA and QUIET states as compared to the terrestrial counterpart.

IDLE state, a source node goes to CONTEND state when it has packet to send. Upon timer expiry in CONTEND state, the source node transmits a RTS, and transits to WFCTS state. The source node waits for returning CTS from its intended receiver and sets its timer to $2\tau_{\max} + T_{\text{cts}}$, where $\tau_{\max}$ is the maximum propagation delay, and T_{cts} is the CTS duration. Similarly, after the intended receiver returns the CTS to the source node, the receiver node goes to WFDATA state and sets its timer to $2\tau_{\max}+T_{\text{data}}$, where T_{data} is the data packet duration. To avoid packet collision, every neighboring node is required to stay in QUIET state upon overhearing an xRTS or xCTS packet. Depending on the overheard control packet, a neighboring node shall set its silent duration to either QUIET_RTS or QUIET_CTS. MACA-U's timing diagram is shown in Fig. 3.1. Note that the curly arrow indicates that a node releases itself from the current handshake.

3.3.1 MACA-U State Transition Rules

MACA-U consists of state transition rules adapted from the terrestrial MACA. Specifically, the modified state transition rules of MACA-U are summarized in Table 3.1. According to the formal specification described by MACAW [14] and FAMA [15] for terrestrial MACA, the deferral rule has a higher order of precedence over the control and timeout rules; that is, when a node in terrestrial MACA overhears any xRTS or xCTS packet, it transits directly to the QUIET state. In contrast, a long propagation delay (i.e., $T_{\text{rts}} \ll \tau_{\max}$ or $T_{\text{cts}} \ll \tau_{\max}$) often causes a node to receive packets other than the intended CTS reply (the CTS packet that originated from the intended receiver node in current handshake, e.g., node "C" in

Table 3.1: State Transition Rules of MACA-U

State \ Event Type	Receives RTS	Receives CTS	Overhears xRTS	Overhears xCTS	Timer Expired
IDLE	Transmit: CTS WFDATA	Disregard packet	Set Timer: QUIET_RTS QUIET	Set Timer: QUIET_CTS QUIET	–
CONTEND	Transmit: CTS WFDATA	Disregard packet	Set Timer: QUIET_RTS QUIET	Set Timer: QUIET_CTS QUIET	Transmit: RTS Set Timer: QUIET_RTS WFCTS
WFCTS	Disregard packet	Decrement BEB Transmit: DATA IDLE	Disregard† packet	Set Timer: QUIET_CTS QUIET	Increment BEB Backoff for Retransmit IDLE
WFDATA	Disregard packet	Disregard packet	Disregard† packet	Disregard† packet	– IDLE
QUIET	Disregard packet	Disregard packet	QUIET*	QUIET*	– IDLE

† In the terrestrial MACA, these three cells transit to QUIET state.

NOTE: QUIET_RTS $= 2\tau_{\max} + T_{\text{cts}}$
QUIET_CTS $= 2\tau_{\max} + T_{\text{data}}$
QUIET* duration $= \max\{Q_{\text{lo}}, Q_{\text{ov}}\}$

Fig. 3.1) or DATA, during WFCTS and WFDATA states, respectively. Therefore, we propose the following state transition rule modifications to improve MACA-U's throughput efficiency (refer to the shaded cells in Table 3.1).

1. In WFCTS state, a source node employs a persistent waiting strategy for the expected CTS. The source node disregards any RTS or xRTS packet. However, the persistent waiting strategy is abandoned when it overhears an xCTS; the source node goes to QUIET state.

2. In WFDATA state, a receiver node employs a persistent waiting strategy for the incoming DATA. The receiver node disregards any RTS, CTS, xRTS and xCTS.

3. In QUIET state, a node remains in QUIET state for an extended period when it overhears xRTS or xCTS. The node computes $\max\{Q_{\text{lo}},Q_{\text{ov}}\}$, where Q_{lo} is the local quiet duration, and Q_{ov} is the overheard control packet's quiet duration. The node shall stay in QUIET state corresponding to the larger of these two variables.

The above state transition rules cater for some scenarios that are much more likely to occur in underwater networks. In the first modification, while a source node resides in WFCTS state, it is reasonable to employ persistent waiting

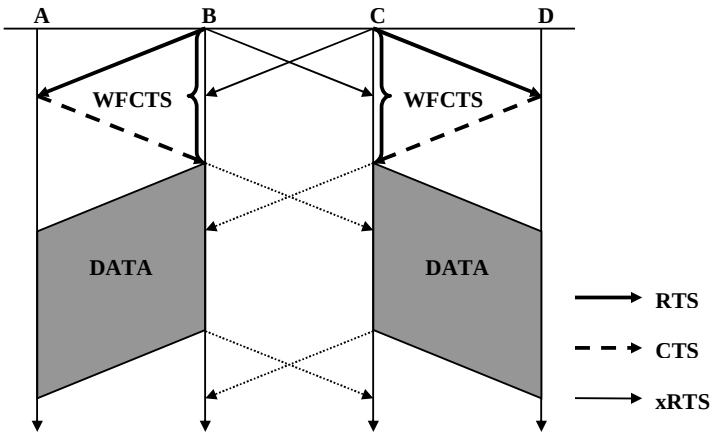


Figure 3.2: Throughput is improved by allowing concurrent transmission at node B and C; a node disregards overheard xRTS in WFCTS state.

strategy for the expected CTS. As can be seen in Fig. 3.2, two neighboring source nodes transmit the RTS packets at around the same time. In this scenario, by allowing the source node to disregard any overheard xRTS during the WFCTS state, the system throughput can be improved due to the concurrent transmission in the neighborhood. In contrast, terrestrial MACA always prioritizes the deferral rule upon overhearing any xRTS or xCTS, i.e., a node transits to QUIET state, and defers its transmission. If we were to follow strictly with the terrestrial MACA's state transition rules, both source nodes shall transit to QUIET state upon overhearing xRTS. Therefore, both nodes waste their data transmission opportunities. However, an exception to the persistent waiting strategy occurs when the source node overhears an xCTS while it is in WFCTS state. In this scenario, the source node shall transit to QUIET state, and abort its data transmission. As can be seen in Fig. 3.3, a potential data collision is very likely to occur at node B, if node C were to transmit its data packet after persistently waiting for node D's CTS. Therefore, by deferring the data transmission at node C, the potential data collision at node B can easily be avoided. In the second modification, it is reasonable to employ persistent waiting strategy for the expected data packet during WFDATA state, as a successful RTS-CTS handshake has already been established. More specifically, a node shall disregard any control packet received while it is in WFDATA state. For example, node B disregards the overheard xRTS, and persistently waits for the expected data packet (Fig. 3.3). Lastly, a node may overhear xRTS or xCTS while it is deferring its data transmission in the QUIET state. In this scenario, a node shall consider the overheard control packet's quiet duration, and extend its quiet duration if the overheard control packet requires a longer silent duration.

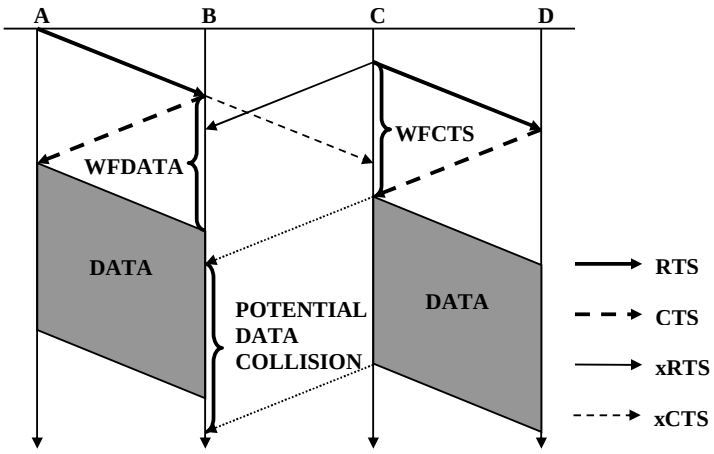


Figure 3.3: Potential data collision is avoided by deferring transmission at node C; a node transits to QUIET state after it overhears xCTS in WFCTS state.

3.3.2 MACA-U's Packet Forwarding Strategy

In fully distributed multi-hop networks, each node may act as a relay node to assist a source node in packet forwarding. Any packet drop that occurs in a relay node is costly as the packet has already consumed valuable channel resources to reach this node. To improve the end-to-end throughput, each node maintains two separate First-In-First-Out (FIFO) queues to differentiate two classes of data traffic; one for data originated from the node itself, and the other for relay data. Higher priority is given to the relay data's queue. For instance, an RTS packet that corresponds to a relay data packet is marked by a higher priority flag.

The long propagation delay in underwater makes it more likely to have two ready neighboring nodes transmit RTS successfully towards each other at around the same time. Without the packet priority assignment, both nodes may wait for the WFCTS timer to expire, and retry several times before giving up. Clearly, this is an undesirable event which leads to low throughput, high latency and energy wastage. This is alleviated by the packet priority assignment which is based on traffic classes. If two senders have equal priorities such as both nodes also intend to transmit relayed packets, then ties can be broken based on unique node IDs.

3.3.3 MACA-U's Backoff Algorithm

Backoff algorithm is a collision resolution methodology to minimize packet collision probability. It is employed during packet retransmission or during the initial contention period, when a node has queued packets to be sent. Similar to terrestrial MACA, when a source node does not receive returned CTS in response to its

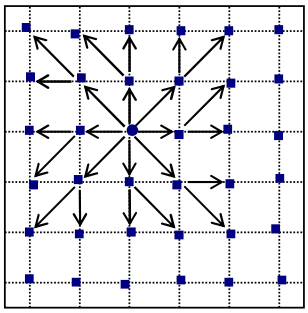


Figure 3.4: The multi-hop network topology used in our simulations.

previous RTS, the source node shall increase its backoff counter, B_{cnt}, according to BEB schemes. Based on the backoff counter value, the node will perform packet retransmission at a later time.

In BEB, each node doubles its backoff counter in the event of RTS failure, and resets its backoff counter to a minimum backoff counter, B_{min}, upon a successful RTS-CTS handshaking. The backoff counter is bounded by a maximum backoff counter, B_{max}. The BEB algorithm can be described by,

$$\begin{cases} B_{\text{cnt}} \leftarrow \min\{2 \times B_{\text{cnt}}, B_{\text{max}}\}, & \text{upon collision,} \\ B_{\text{cnt}} \leftarrow B_{\text{min}}, & \text{upon successful transmission.} \end{cases} \tag{3.1}$$

In underwater networks, the retransmission or contention slot duration is defined by $T_{\text{rts}} + \tau_{\text{max}}$. The backoff interval, T_{bk}, of a node can be expressed as

$$T_{\text{bk}} = \mathit{uniform}\{0, B_{\text{cnt}}\} \times (T_{\text{rts}} + \tau_{\text{max}}). \tag{3.2}$$

If the large propagation delay is not considered in defining the slot length, it may lead to a more aggressive contention, and result in higher packet collision rate. In our work, we compare the BEB's throughput performance against Multiplicative Increase Linearly Decrease (MILD) [14], Exponential Increase Exponential Decrease (EIED) [53], Linearly Increase Linearly Decrease (LILD) [54], and IEEE 802.11-alike rotating backoff [16].

3.4 Simulations And Results

3.4.1 Simulation Model

In our simulation, the multi-hop network topology comprised of 36 static nodes with a grid spacing of 700 m as shown in Fig. 3.4. Instead of precisely placing each node at the grid intersection point, we introduce some degree of randomness by

allowing each node to deviate from the grid intersection point by a maximum of 10% of its grid spacing, in both vertical and horizontal directions. The maximum transmission range for each node is 1.75 times the grid spacing, or 1225 m in our simulation topology. Hence, each node has exactly 8 one-hop neighboring nodes and 16 two-hop neighboring nodes. A wrap-around strategy is applied to all boundary nodes in order to distribute network load evenly and eliminate boundary effect. For every packet generated by each node, it randomly selects one of its 16 two-hop neighbors as an end destination with equal probability. In the figure, we only show the static routing pattern for one node (the round node). Every other node in the network topology assumes the same static routing pattern. All nodes are assumed to be equipped with a half-duplex omni-directional antenna. The bit rate of each node is assumed to be 2400 bps. The acoustic propagation speed is 1500 m/s. Every node operates independently of each other and traffic load is divided evenly among all nodes according to the Poisson distribution. The channel is assumed to be error-free. Thus, packet losses are contributed by packet collision. For MACA-U simulations, all control packets' lengths are 100 bits, i.e., RTS and CTS are of equal length in our simulation. Data packet lengths of 1200, 2400, 4800 bits have been simulated. For BEB backoff parameters, $B_{\min}$ is 1, and $B_{\max}$ is 64. Each node maintains two separate FIFO buffers for every one-hop neighbor with maximum size of 100 packets. There is no ACK involved in our simulations. To avoid transient effect, results are collected from $200,000$ s to $1,000,000$ s.

3.4.2 Simulation Results

The simulation's objective is to study MACA-U's performance, specifically on its throughput in underwater networks. Note that the throughput per node is a unitless metric, as it has been normalized to single-hop channel capacity, i.e., 2400 bps. As presented in Fig. 3.5, we benchmark our MACA-U against the conventional pure Aloha protocol. We observe that the pure Aloha scheme only has a maximum throughput per node of $0.0080 - 0.0085$. In addition, its throughput per node drops as the offered load per node increases beyond its maximum throughput operating point. This is reasonable as pure Aloha does not deploy any collision avoidance mechanism in the presence of hidden nodes.

In contrast, MACA-U maintains its stable throughput as the offered load increases, at the expense of small communication overhead (exchanges of control

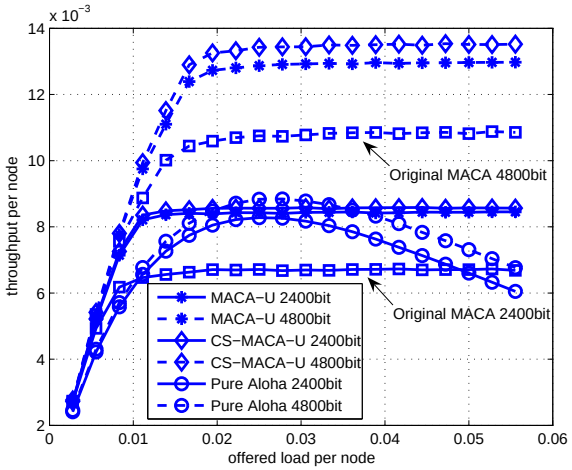


Figure 3.5: Throughput comparison for MACA-U, CS-MACA-U, pure Aloha and original MACA.

packets). The stabilized throughput characteristic can be explained by its efficient collision avoidance mechanism, and collision resolution mechanism in packet retransmission. In the presence of hidden nodes, the probability of data packet collision at higher offered load is minimized by prior exchange of short RTS-CTS packets, which are used for virtual carrier sensing. Next, we simulated an improved carrier sensing variant, CS-MACA-U. This protocol performs physical state verification at the time instant when it wishes to transmit its RTS. The node only transmits when its physical state is IDLE. Otherwise, it shall perform backoff and try to access the channel again. By performing physical state verification, the protocol eliminates potential data packet collision which may have occurred in packet sensing. For instance, data collision may result if a source node happens to transmit its own RTS packet even though it is in the middle of receiving an incoming RTS packet. Lastly, we compare the performance of MACA-U against the original MACA protocol (without the state transition rules adaptation). For both simulated packet lengths, MACA-U outperforms the original MACA by around 20% for the saturation throughput. Note that the original MACA protocol for 2400 bits actually has a lower throughput per node than the pure Aloha counterpart at the offered load region of $0.01 - 0.05$.

Next, we study the effects of data packet sizes in MACA-U with the assumption of fixed control packet size. As illustrated in Fig. 3.6, MACA-U exhibits a stable throughput per node for all ranges of data packet sizes. It is not surprising that the throughput increases as we send a larger data packet upon a successful handshake. The observation suggests that MACA-U may be improved using a packet-train concept. Obviously, it is very costly to only send a single data

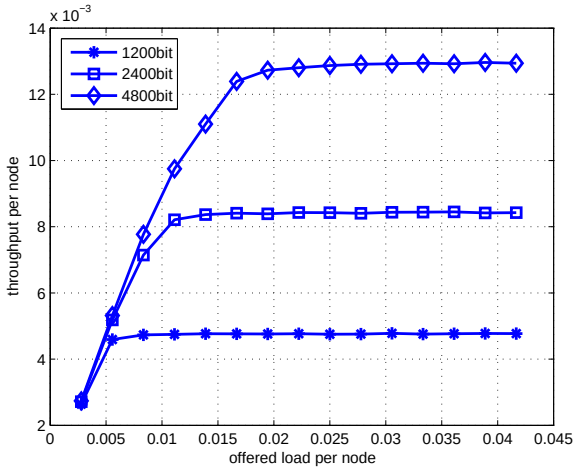


Figure 3.6: Effects of different data packet sizes.

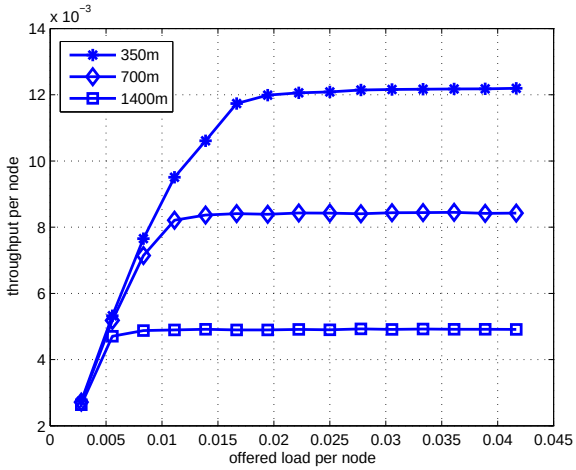


Figure 3.7: Effects of different grid sizes.

packet upon a successful handshake.

Fig. 3.7 shows the effects of grid sizes when the data length is 2400 bits. As shown, MACA-U is rather sensitive to the inter-node distances. The explanation for the lower throughput as separation distance increase is due to a larger propagation delay in the exchange of RTS-CTS packets. This suggests that handshaking-based protocol is more appropriate in short range multi-hop underwater networks.

Fig. 3.8 illustrates the performance of various backoff strategies in MACA-U. For EIED and LILD schemes, we have simulated various combinations of factors, i.e., additive increase or decrease factors, multiplicative increase or decrease factors. Here, we only present the optimal cases. As opposed to terrestrial MACAW, MILD actually performs worse than BEB in terms of throughput efficiency. The reason is that MILD requires more time steps in the decrement back to a smaller contention window. We observed that a better throughput is achieved when the backoff algorithm assumes a drastic decrement, i.e., BEB, EIED. However, we

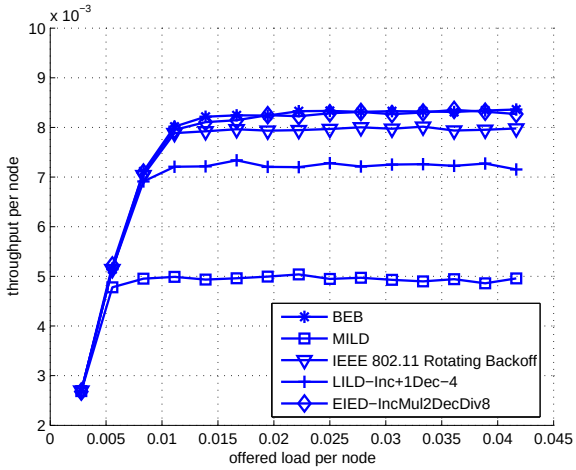


Figure 3.8: Performance evaluation of various collision resolution schemes.

point out that fairness remains as an issue that needs to be solved.

3.5 Summary

In this chapter, we have proposed and studied the MACA-U protocol, which is an adaptation of the original MACA protocol (i.e., 3-way RTS/CTS/DATA handshaking-based MAC with BEB backoff algorithm) for use in multi-hop UWA networks. The operation rules of terrestrial MACA are modified, so as to better handle potential problematic scenarios that only arise due to the long propagation delay. Three areas of improvement are identified: (i) state transition rules, (ii) packet forwarding strategy, and (iii) backoff algorithm, and modified accordingly by taking the long propagation delay into account. From our simulation results, we have demonstrated that the adapted MACA-U improves throughput performance for all offered load ranges, as compared to the original MACA that applied blindly into underwater networks. Compared to pure Aloha that does not employ any collision avoidance mechanism, MACA-U can maintain a stable throughput as the offered load further increases. As opposed to the terrestrial MACAW's finding, we also show that the MILD backoff algorithm actually performs worse than BEB in terms of throughput, in high latency networks. Due to its simplicity and throughput stability, the adapted MACA-U can serve as a more appropriate reference MAC protocol, which a future, more advanced underwater MAC may benchmark its performance against. In our following Chapter 4 and Chapter 5, we will adopt MACA-U as a benchmark protocol to gauge the performance improvement attained by our proposed MAC protocols.

Chapter 4

A MAC Protocol with Bidirectional-Concurrent Packet Exchange

4.1 Introduction

From Chapter 2, we find that most of the existing sender-initiated handshaking-based MAC protocols [9, 10, 18–21, 23–25, 28, 30] often adopt a unidirectional data transmission approach, in which an initiating sender is allowed to transmit either a single or multiple data packets back-to-back to its intended receiver upon each successful handshake. Thus, coupled with the multi-way control packet exchange in the presence of long propagation delays, means that the channel utilization is usually poor, which in turn greatly impacts both throughput and delay performance. Although the packet train approach can improve the performance to some extent, the data transmission process is still inefficient due to high latency overhead. More importantly, this unidirectional packet train approach fails to take full advantage of the opportunities provided by the underwater acoustic channel.

In this chapter, we present a novel approach to improve low channel utilization, by exploiting the slow propagation speed of sound in water. Our idea is inspired by the key observation that two nodes may transmit to each other at around the same time, and yet, not result in any data collision in underwater networks. In this case, the transmission duration of each data packet should be much shorter than the inter-nodal propagation delay; this condition is often easily satisfied in underwater communications. We then propose the idea of

"bidirectional, concurrent data packet exchange" to optimize the data transmission efficiency, in which a sender-receiver (S-R) node pair is allowed to transmit data packets to each other for every successful handshake. This bidirectional transmission should occur in a concurrent manner so that the transmissions are tightly packed. To further amortize the high latency overhead, we also present a *"packet bursting"* idea that allows the S-R node pair to exchange multiple rounds of bidirectional packet transmissions. This is different from the normal packet train approach; in our approach, the entire set of data packets are actually transmitted over several discontinuous packet bursts.

Based on the above ideas, we propose a novel single-channel handshaking-based MAC protocol, which we call BiC-MAC (Bidirectional-Concurrent MAC with packet bursting). Although its handshake is sender-triggered, the intended receiver can exploit this opportunity to initiate potentially multiple rounds of concurrent, bidirectional data exchange when it has data packets in return. Using this approach, both nodes in the S-R node pair share only one set of communication overhead, e.g., backoff time, delay incurred by handshake, etc., and the proportion of time spent on control signaling is significantly reduced. Furthermore, the data packet transmission efficiency is greatly enhanced because both nodes can access the channel simultaneously after the floor reservation. Note that a versatile MAC framework is conceived (as described in Section 4.3.1), so that BiC-MAC can operate in three possible bidirectional transmission modes to cater for the fact that the S-R node pair may not intend to exchange the same number of data packets. It is also worthwhile to mention that BiC-MAC does not need clock synchronization. Finally, if the receiver does not have any data packets in return, normal unidirectional data packet transmissions can still be performed. In short, MACA [13] with unidirectional packet train is a special case of BiC-MAC. To the best of our knowledge, BiC-MAC is the first handshaking-based MAC protocol that utilizes a comprehensive bidirectional, concurrent transmission MAC framework for exchanging data packet bursts in underwater acoustic networks.

The remainder of this chapter is organized as follows. We first describe the system model in Section 4.2. Then, the BiC-MAC protocol details are presented in Section 4.3. Next, we evaluate BiC-MAC using both multi-hop and single-hop networks, and their simulation results are discussed in Section 4.4 and Section 4.5, respectively. Finally, we conclude this chapter in Section 4.7.

4.2 System Model

We consider an ad hoc, static, multi-hop acoustic network, in which each node is equipped with a single omni-directional, *half-duplex* underwater acoustic modem. However, the "static" nodes still have some limited degree of movement as they are typically anchored to the sea-bed, and subjected to a maximum sway distance caused by underwater currents. As for the MAC protocol, all nodes exchange both control and data packets in a single channel. Currently, we only consider fixed-length packets. We also do not put any upper limit on the number of retries when RTS attempts fail. We assume that every node knows its estimated propagation delay from each of its one-hop neighbors (we will explain how to relax this assumption in Section 4.3.1.3). The estimations can be performed during network initialization. As an example, a node x could randomly select a time to broadcast a control packet to its one-hop neighbors. Upon hearing the control packet, a neighbor node y records the arrival time, and then randomly chooses a time to reply with a control packet. The node y specifies within the control packet the time difference between the previous packet's arrival time, and the current packet's transmission time. After receiving the control packet, the node x could compute their inter-nodal propagation delay by measuring the round-trip time (RTT). Note that any estimation errors and the fluctuations caused by sway movements are typically very small compared to the propagation delays; hence, their effects could be easily accommodated with the use of guard times.

4.3 The BiC-MAC Protocol

4.3.1 How the BiC-MAC Protocol Works

Table 4.1 defines the notation used in the protocol description. For simplicity, in each handshake, we only allow the total number of data packets transmitted by an intended receiver node to be at most the same as that of the initiating sender, i.e., $n_{\mathrm{R}} \leq n_{\mathrm{S}}$. With this assumption, the bidirectional packet transmission behavior of BiC-MAC is limited to three possible scenarios, as shown by the timing diagrams in Fig. 4.1(a)–4.1(c).

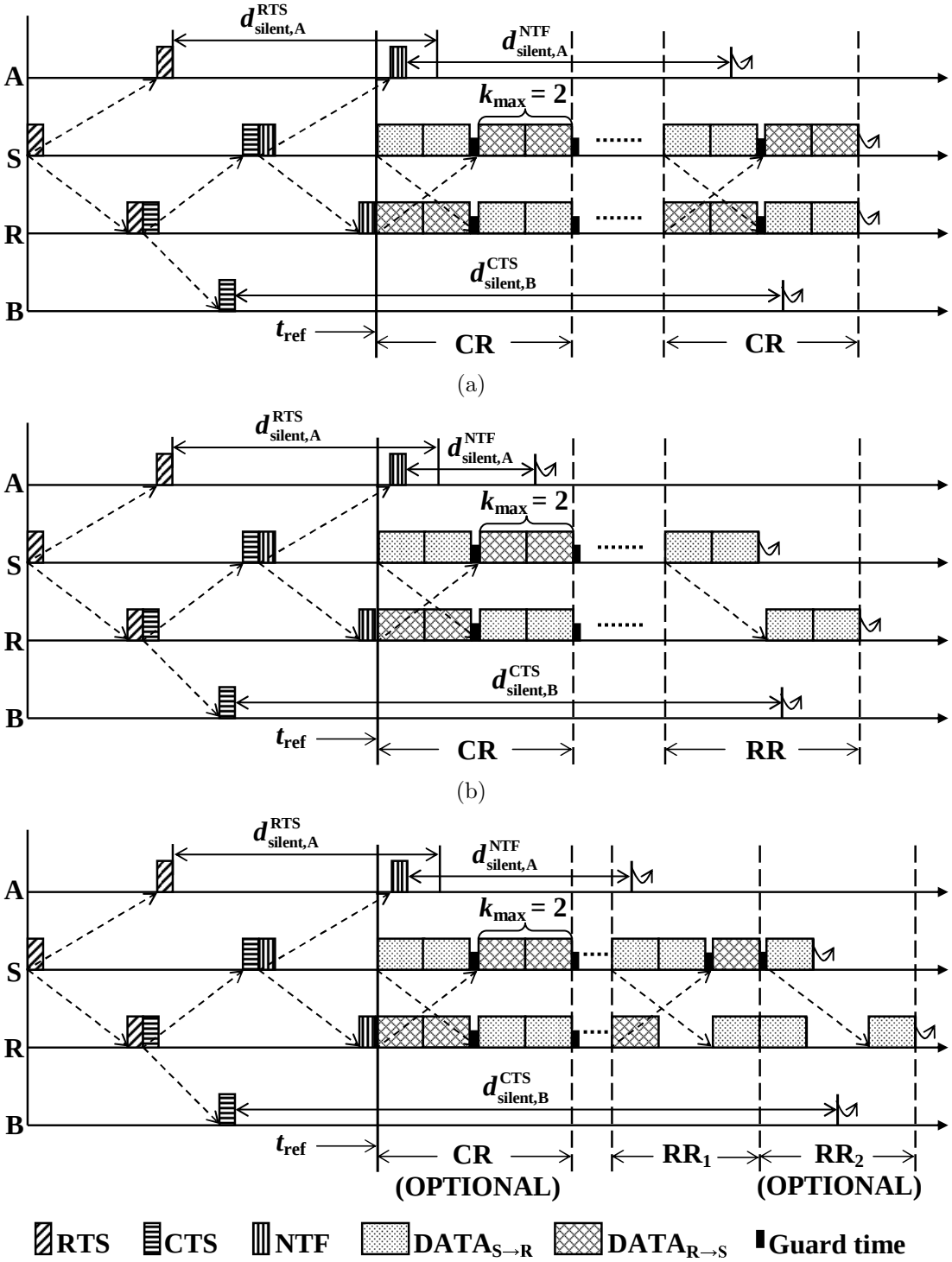


Figure 4.1: Timing diagrams of BiC-MAC: (a) Type 1 scenario, (b) Type 2 scenario, (c) Type 3 scenario. Nodes "S" and "R" refer to the initiating sender and its intended receiver, respectively; nodes "A" and "B" are the one-hop neighbors of nodes "S" and "R", respectively. Small constant guard times can be used to accommodate any propagation delay's estimation error, as well as any slight difference in the perception of t_{ref} by both "S" and "R". The releases from the current handshake are shown as curly arrows. Note that BiC-MAC is equipped with three operating modes, due to an S-R pair may not intend to exchange the same amount of data packets. Nonetheless, these three scenarios have identical sequences of control packet exchanges.

Table 4.1: Notation used for explaining the BiC-MAC protocol

Notation	Description
T_{DATA}	Transmission time of each fixed-length data packet
T_x	Transmission time of each fixed-length control packet of type x, where $x \in \{\text{RTS}, \text{CTS}, \text{NTF}\}$
$\tau_{\max}$	Maximum propagation delay that corresponds to a node's maximum transmission range
$\tau_{\max,i}$	Maximum propagation delay between node i and its one-hop neighbors
$\tau_{i,j}$	Inter-nodal propagation delay between node i and node j
S_{burst}	Threshold number of accumulated data packets for triggering an RTS attempt
$T_{\max}$	Time threshold for triggering an RTS attempt
t_{ref}	Starting reference time for the S-R node pair's bidirectional data exchange
$d^x_{\text{silent},i}$	Duration that node i must remain silent after overhearing a type x control packet, where $x \in \{\text{RTS}, \text{CTS}, \text{NTF}\}$
$d_{\text{busy},x}$	Busy duration information that is carried within a type x control packet, where $x \in \{\text{RTS}, \text{CTS}, \text{NTF}\}$
$d^{\text{rx}}_{\text{busy},i}$	Busy duration that node i spent from t_{ref} until it finishes receiving the last intended data packet
$d_{\text{toc},i}$	Time-of-Completion; duration that node i spent from t_{ref} until it is released from the handshake, i.e., upon completing its bidirectional data transmissions/receptions
T_{guard}	Guard time to accommodate any inter-nodal propagation delay's estimation error, and the transceiver's transmit-receive turnaround time
k_i	Number of data packets that node i transmits either in a Complete Round (CR) or a Residual Round (RR)
$k_{\max}$	Maximum number of data packets that a node is allowed to transmit in a CR so as to avoid TX-RX collisions
n_{CR}	Total number of CRs required in the current bidirectional data exchange
d_{CR}	Duration of a single CR in the current bidirectional data exchange
n_i	Total number of data packets (both relayed and new) that node i intends to transmit in current handshake
$n_{x,i}$	Number of data packets of type x that node i intends to transmit in current handshake, where $x \in \{\text{relay}, \text{new}\}$
$\mathcal{N}_i$	Set of node i's one-hop neighboring nodes

4.3.1.1 Channel Reservation Phase

As can be seen, the three possible scenarios have identical sequences of control packet exchanges. In BiC-MAC, an idle node adopts a hybrid of "batch-by-size" and "batch-by-time" strategies to determine when to trigger its RTS attempt. After the triggering condition has been satisfied, the sender shall initiate its contention timer according to the Binary Exponential Backoff (BEB) algorithm, and then broadcast an RTS packet to its one-hop neighbors upon the timer expiry (see Section 4.3.2). In the RTS packet, the sender announces its intended total number of data packets to be transmitted for the current handshake, n_{S}. Since a node may transmit both relayed and self-originated traffic, n_{S} can be specified using two separate fields, $n_{\mathrm{relay,S}}$ and $n_{\mathrm{new,S}}$.

Upon receiving the RTS packet, a receiver checks whether it has any data packets in return, before responding with a CTS packet. Note, however, that the intended receiver node may only transmit the CTS response provided it is currently not involved in any other handshake, and is also not required to remain silent. In the scenario where the receiver is interested in bidirectional data packet transmissions, it shall announce the total number of data packets that it wishes to transmit, n_{R}, in the CTS packet (or both $n_{\mathrm{relay,R}}$ and $n_{\mathrm{new,R}}$). As long as $n_{\mathrm{R}} > 0$, the condition is sufficient for the bidirectional data exchange. Hence, the CTS packet functions like an RTS packet where a receiver can invite the initiating sender for bidirectional transmissions. On the other hand, if the receiver node does not have any data packets in return, both $n_{\mathrm{relay,R}}$ and $n_{\mathrm{new,R}}$ fields are set to zero. Consequently, the handshake reduces to the conventional 3-way handshake, i.e., MACA with unidirectional packet train. Since the concurrent, bidirectional data exchange requires the participating nodes (S-R node pair) to cycle between packet transmissions and receptions, the receiver must also compute its data transmission/reception schedule. We will explain how the schedule can be computed in the following section, but it should be noted for now that both participating nodes need to be aware of when and how many data packets can be exchanged in order to avoid transmit-receive (TX-RX) collisions. Finally, a sender (receiver) node also locally computes its expected busy duration, i.e., $d_{\mathrm{busy,RTS}}$ ($d_{\mathrm{busy,CTS}}$), and includes it in the broadcasted RTS (CTS) packet. Any neighboring node that overhears the control packet shall extract the embedded busy duration, and then locally compute its silent duration. Subsequently, it will defer initializing

any transmission so as to avoid packet collisions with the ongoing transmissions.

Upon receiving the CTS and validating that $n_{\mathrm{R}} > 0$, the initiating sender will compute its data transmission schedule as well. In our protocol, a new type of control packet is introduced, namely, the Notification (NTF) packet. The functions of an NTF packet are two-fold. Firstly, a sender can include its busy duration, $d_{\mathrm{busy,NTF}}$, in the control packet, so that its first-hop neighbors, $\mathcal{N}_{\mathrm{S}}$, could compute their respective silent durations. As illustrated in Fig. 4.1(a)–4.1(c), the sender's first-hop neighbors (e.g., node "A") should extend their silent durations upon overhearing an NTF packet, so that the sender can successfully receive its incoming bidirectional data packets. Secondly, the NTF packet synchronizes both sender and receiver to a common starting reference time, t_{ref}, such that their bidirectional packet bursts exchange can be performed in a concurrent manner[1]. Specifically, after transmitting an NTF, the sender will wait for a duration of $\tau_{\mathrm{S,R}}$ before it starts to transmit its data packets. In contrast, the receiver can start its data transmission immediately after receiving the NTF packet. Note that small guard times can be inserted as shown so as to accommodate any slight difference in the perception of t_{ref} by both participating nodes (to be detailed in Section 4.3.1.2).

As mentioned, the purpose of the busy durations is to inform all neighboring nodes of the S-R node pair so that they can defer their transmissions accordingly. This is a vulnerable period where any packet transmissions from these neighboring nodes may interfere with the ongoing bidirectional transmissions. In the following, we first show how an S-R node pair can compute their respective busy durations:

$$\begin{cases} d_{\mathrm{busy,RTS}} = T_{\mathrm{CTS}} + 2\tau_{\mathrm{S,R}} + T_{\mathrm{NTF}} + \tau_{\mathrm{max,S}}, \\ d_{\mathrm{busy,CTS}} = 2\tau_{\mathrm{S,R}} + T_{\mathrm{NTF}} + d^{\mathrm{rx}}_{\mathrm{busy,R}}, \\ d_{\mathrm{busy,NTF}} = \tau_{\mathrm{S,R}} + d^{\mathrm{rx}}_{\mathrm{busy,S}}, \end{cases} \tag{4.1}$$

where $\tau_{\mathrm{max,S}} = \max_{j \in \mathcal{N}_{\mathrm{S}}}(\tau_{\mathrm{S},j})$, and $d^{\mathrm{rx}}_{\mathrm{busy},i}$ is the busy duration that a participating node i spends from t_{ref} until it finishes receiving the last intended data packet (the computation will be detailed in Section 4.3.1.2).

Upon overhearing a control packet and extracting the busy duration it carries, every neighbor can compute its silent duration locally. This is similar to the

[1]Note that BiC-MAC does not need clock synchronization. Both sender and receiver merely have a common understanding of the starting time of the bidirectional packet exchange, relative to their own respective *local* clocks.

concept of IEEE 802.11's Network Allocation Vector (NAV) [16] in terrestrial wireless networks, except for the following difference. To optimize the performance, the silent duration does not need to be as long as the busy duration, because of the inter-nodal propagation delays between the participating nodes and their overhearing neighbors. Each neighbor only needs to ensure that any transmission after its silent duration will not interfere with the ongoing transmissions. Hence, the respective silent durations can be computed as

$$\begin{cases} d^{\mathrm{RTS}}_{\mathrm{silent},i} = d_{\mathrm{busy,RTS}} - \tau_{\mathrm{S},i}, & i \in \mathcal{N}_{\mathrm{S}}, \\ d^{\mathrm{CTS}}_{\mathrm{silent},j} = d_{\mathrm{busy,CTS}} - 2\tau_{\mathrm{R},j}, & j \in \mathcal{N}_{\mathrm{R}}, \\ d^{\mathrm{NTF}}_{\mathrm{silent},i} = d_{\mathrm{busy,NTF}} - 2\tau_{\mathrm{S},i}, & i \in \mathcal{N}_{\mathrm{S}}. \end{cases} \tag{4.2}$$

Note that a successful NTF packet's reception at the sender's first-hop neighbors is imperative to ensure that an uninterrupted bidirectional transmission can be performed. Hence, the $d^{\mathrm{RTS}}_{\mathrm{silent},i}$ duration needs to be sufficiently large to cover till the time instant when the sender's most distant neighbor could completely overhear the NTF packet. In this manner, the neighboring nodes' silent durations can be extended via overhearing the subsequent NTF packet, if any.

4.3.1.2 Bidirectional-Concurrent Data Transmission Phase

As mentioned, the bidirectional transmissions in BiC-MAC can be categorized into three possible scenarios, namely, Type 1, Type 2, and Type 3. The reason we conceive this versatile MAC framework is because it is likely that both sender and receiver may not intend to transmit the same amount of data packets. To better facilitate our explanation, we introduce the terms *Complete Round (CR)* and *Residual Round (RR)*.

Definition 1 *A Complete Round (CR) is defined as the time window over which both nodes in the S-R node pair transmit the maximum allowable number of data packets, k_{max}, to each other through a concurrent, bidirectional data packet transmission approach.*

Definition 2 *A Residual Round (RR) is defined as the time window over which either (i) only one of the nodes in a S-R node pair transmits its data packets, or, (ii) both nodes transmit to each other through a concurrent, bidirectional data packet transmission approach, but at least one of the nodes transmits less than k_{max} data packets.*

To improve BiC-MAC's reliability, a small constant guard time, T_{guard}, can be inserted every time a node switches between transmit and receive modes during the data transmission phase; this guard time will accommodate for: (i) any estimation error in the inter-nodal propagation delays, (ii) transceiver's TX-RX turnaround time, (iii) maximum sway distance caused by underwater currents, and (iv) any slight difference in the S-R node pair's perception of starting reference time, t_{ref}. In general, the required guard time is typically very small (in the order of tens of milliseconds [55]) compared to the data packet's transmission time. It is worthwhile to mention that the use of bidirectional transmission allows BiC-MAC to correct any possible error in the inter-nodal delay estimates; specifically, a sender can examine an incoming data packet's arrival time in a CR, and use the deviation from the expected arrival time for the correction.

To enhance efficiency, the bidirectional transmissions' time-of-completion of the S-R pair should be minimized, subject to the constraint that TX-RX collisions do not occur. In each CR, as well as any RR that has concurrent, bidirectional transmission, the condition to avoid TX-RX collisions in each round is

$$k_i T_{\text{DATA}} + T_{\text{guard}} \leq \tau_{\text{S,R}}, \quad i \in \{\text{S}, \text{R}\}. \tag{4.3}$$

From (4.3), it is straightforward to compute the maximum allowable number of data packets to be sent in each CR using

$$k_{\max} = \left\lfloor \frac{\tau_{\text{S,R}} - T_{\text{guard}}}{T_{\text{DATA}}} \right\rfloor. \tag{4.4}$$

To pack the transmissions as tightly as possible, the participating nodes must concurrently transmit $k_i = k_{\max}$ data packets in every CR. Thus, the S-R node pair shall expect identical CR duration in all three scenarios, i.e., $d_{\text{CR}} = \tau_{\text{S,R}} + k_{\max} T_{\text{DATA}}$. Fig. 4.1(a)–4.1(c) show the example scenarios when $k_{\max} = 2$. The total number of CRs required for the current handshake, n_{CR}, can be found using

$$n_{\text{CR}} = \lfloor \min(\alpha_{\text{S}}, \alpha_{\text{R}}) \rfloor, \quad \text{where } \alpha_i = n_i / k_{\max}, i \in \{\text{S}, \text{R}\}. \tag{4.5}$$

When $n_{\text{R}} \leq n_{\text{S}}$, it can be further reduced to $n_{\text{CR}} = \lfloor n_{\text{R}} / k_{\max} \rfloor$.

Next, we describe the three possible scenarios that arise in BiC-MAC:

Type 1: This scenario is characterized by the presence of at least one CR, and no RR. Here, $n_{\mathrm{S}} = n_{\mathrm{R}}$. The time-of-completion and the busy duration are

$$\begin{aligned} d_{\mathrm{toc},i} &= d^{\mathrm{rx}}_{\mathrm{busy},i} \\ &= n_{\mathrm{CR}} d_{\mathrm{CR}} + (n_{\mathrm{CR}} - 1) T_{\mathrm{guard}}, \quad i \in \{\mathrm{S}, \mathrm{R}\}. \end{aligned} \tag{4.6}$$

Type 2: This scenario is characterized by the presence of at least one CR, and a single RR. Here, $n_{\mathrm{S}} > n_{\mathrm{R}}$. In the RR, the sender shall transmit its remaining data packets unidirectionally. The time-of-completion and the busy duration are

$$\begin{cases} d^{\mathrm{rx}}_{\mathrm{busy,S}} = n_{\mathrm{CR}}(d_{\mathrm{CR}} + T_{\mathrm{guard}}), \\ d_{\mathrm{toc,S}} = d^{\mathrm{rx}}_{\mathrm{busy,S}} + (n_{\mathrm{S}} - n_{\mathrm{CR}} k_{\max}) T_{\mathrm{DATA}}, \\ d_{\mathrm{toc,R}} = d^{\mathrm{rx}}_{\mathrm{busy,R}} = d_{\mathrm{toc,S}} + \tau_{\mathrm{S,R}}. \end{cases} \tag{4.7}$$

Type 3: This scenario is characterized by the presence of optional CRs, and a single RR. Unlike Type 2 scenario, the RR here is further divided into two sub-residual rounds – a mandatory RR_1 and an optional RR_2. Therefore, the bidirectional transmissions in this scenario can be any one of the following configurations: the presence of (i) CR, RR_1, and RR_2, (ii) CR and RR_1, (iii) RR_1 and RR_2, or (iv) RR_1 only. The optional RR_2 is present only when a sender node has $n_{\mathrm{S}} > (n_{\mathrm{CR}} + 1) k_{\max}$ number of data packets to be transmitted. Notice that a receiver always transmits $k_{\mathrm{R}} < k_{\max}$ data packets in RR_1. As soon as the sender finishes receiving its data packets in RR_1 (after T_{guard}), it shall transmit its excess data packets unidirectionally in RR_2, if any. Again, note our assumption that, $n_{\mathrm{S}} \geq n_{\mathrm{R}}$. The time-of-completion and the busy duration can be expressed as

$$\begin{cases} d^{\mathrm{rx}}_{\mathrm{busy,S}} = n_{\mathrm{CR}}(d_{\mathrm{CR}} + T_{\mathrm{guard}}) + (n_{\mathrm{R}} - n_{\mathrm{CR}} k_{\max}) T_{\mathrm{DATA}} \\ \qquad\qquad + \tau_{\mathrm{S,R}}, \\ d_{\mathrm{toc,S}} = \begin{cases} d^{\mathrm{rx}}_{\mathrm{busy,S}} + (n_{\mathrm{S}} - (n_{\mathrm{CR}} + 1) k_{\max}) T_{\mathrm{DATA}} \\ + T_{\mathrm{guard}}, & \text{if } \mathrm{RR}_2 \text{ is present}, \\ d^{\mathrm{rx}}_{\mathrm{busy,S}}, & \text{otherwise}, \end{cases} \\ d_{\mathrm{toc,R}} = d^{\mathrm{rx}}_{\mathrm{busy,R}} \\ \qquad = \begin{cases} d_{\mathrm{toc,S}} + \tau_{\mathrm{S,R}}, & \text{if } \mathrm{RR}_2 \text{ is present}, \\ n_{\mathrm{CR}}(d_{\mathrm{CR}} + T_{\mathrm{guard}}) + (n_{\mathrm{S}} - n_{\mathrm{CR}} k_{\max}) T_{\mathrm{DATA}} \\ + \tau_{\mathrm{S,R}}, & \text{otherwise}. \end{cases} \end{cases} \tag{4.8}$$

Based on (4.6)–(4.8), a participating node i can deduce its starting times (with respect to t_{ref}) of all the rounds, $\mathbf{s}_i$, as well as the number of data packets that can be transmitted in every round, $\mathbf{k}_i$. For example, a Type 2 scenario's sender may start to transmit in each round at $\mathbf{s}_{\text{S}} = [0, A, \cdots, (n_{\text{CR}} - 1)A, n_{\text{CR}}A]$ with $\mathbf{k}_{\text{S}} = [k_{\max}, k_{\max}, \cdots, k_{\max}, B]$, where $A = d_{\text{CR}} + T_{\text{guard}}$ and $B = n_{\text{S}} - n_{\text{CR}}k_{\max}$; a receiver node, on the other hand, has a transmission schedule of $\mathbf{s}_{\text{R}} = [0, A, \cdots, (n_{\text{CR}} - 1)A]$ with $\mathbf{k}_{\text{R}} = [k_{\max}, k_{\max}, \cdots, k_{\max}]$.

Finally, in order to prevent an S-R node pair from reserving the channel for an unreasonably long duration, each node is permitted to transmit at most S_{burst} data packets for each successful handshake. This also leads to shorter delays. Another issue that may arise in an actual deployment scenario is that, the sender-receiver separation might be too close at times, resulting in $\tau_{\text{S,R}} < T_{\text{DATA}} + T_{\text{guard}}$. When this occurs, it is more beneficial to perform unidirectional packet transmissions, as the communication overheads involved are much smaller.

During the channel reservation phase, an NTF may experience packet collision at its intended receiver, i.e., NTF failure, and thus the intended receiver cannot initiate its bidirectional data transmissions. We include a mechanism for the initiating sender to detect the NTF failure event. When a sender detects that there is no incoming data packet from its intended receiver in the first CR, it will assume that an NTF failure has occurred. Hence, the sender aborts its subsequent transmissions, and releases itself from current handshake. Similarly, the receiver releases from current handshake after its WFNTF (wait-for-NTF) timer expiry.

4.3.1.3 Relaxing the Assumption of First-hop Neighbors' Knowledge

We now explain how this assumption can be relaxed. By utilizing the 3-way RTS/CTS/NTF handshake, it is possible to estimate the inter-nodal propagation delay between a sender and its intended receiver, i.e., $\tau_{\text{S,R}}$. After sending an RTS, a sender can measure the RTT upon receiving the CTS reply, which can be used to compute $\tau_{\text{S,R}}$. Similarly, a receiver can compute the inter-nodal propagation delay via measuring the RTT of the CTS/NTF handshake. Although both nodes may have slightly different estimates of the delay, BiC-MAC can accommodate such estimation errors via its guard time.

By estimating $\tau_{\text{S,R}}$ only during RTS/CTS/NTF handshake, we need to make the following modifications to BiC-MAC. Firstly, instead of computing its

data transmission/reception schedule when an intended receiver prepares its CTS packet, the receiver now needs to defer this computation to the time instant when it finishes receiving the NTF. This is because it requires $\tau_{\mathrm{S,R}}$ in order to construct its schedule, but this information is only available at the end of CTS/NTF handshake. Secondly, the computation of both busy and silent durations need to be changed. We first show how $d_{\mathrm{busy,RTS}}$ and $d_{\mathrm{busy,CTS}}$ in (4.1) can be modified as follows:

$$\begin{cases} d_{\mathrm{busy,RTS}} = T_{\mathrm{CTS}} + T_{\mathrm{NTF}} + 3\tau_{\max}, \\ d_{\mathrm{busy,CTS}} = 2\tau_{\max} + T_{\mathrm{NTF}}. \end{cases} \tag{4.9}$$

At the time a sender (receiver) node prepares its RTS (CTS) packet, $\tau_{\mathrm{S,R}}$ and $\tau_{\mathrm{max,S}}$ remain unknown. As such, in the computation of busy durations for RTS and CTS in (4.1), both $\tau_{\mathrm{S,R}}$ and $\tau_{\mathrm{max,S}}$ will have to be replaced with $\tau_{\max}$ (maximum propagation delay due to a node's maximum transmission range). Note that in the above $d_{\mathrm{busy,CTS}}$, we do not include $d^{\mathrm{rx}}_{\mathrm{busy,R}}$ since this value will only be known when the receiver computes its bidirectional data schedule. Hence, after $d^{\mathrm{rx}}_{\mathrm{busy,R}}$ is known, the receiver can announce the updated busy durations, which can be carried in its data packet's header. As for the NTF busy duration in (4.1), it will remain the same. Finally, the silent duration calculations in (4.2) will now be simply $d^{x}_{\mathrm{silent},i} = d_{\mathrm{busy},x}$, where $x \in \{\mathrm{RTS}, \mathrm{CTS}, \mathrm{NTF}\}$ for any neighboring node i. The longer silent duration arising from the above modifications sacrifice part of BiC-MAC's performance, but it relaxes the need for propagation delay estimation during network initialization.

4.3.2 RTS Attempts and Backoff Algorithm

In a multi-hop network, a node could generate its own data packets, as well as help its immediate neighboring nodes to relay their packets. Here, we assume that each node maintains two separate buffers for each of its one-hop neighbors; one for relayed traffic, and the other for self-originated traffic. We also assume that relayed traffic is prioritized over self-originated traffic because the former has already consumed some network resources previously. The BiC-MAC protocol uses a hybrid of "*batch-by-size*" and "*batch-by-time*" mechanisms to determine when to trigger an RTS attempt[2].

[2]Similar to [56], we utilize queue size and waiting time as a threshold for assembling packet burst.

For the "batch-by-size" mechanism, a sender accumulates at least S_{burst} data packets for a particular neighbor before triggering its RTS attempt. The composition of S_{burst} may consist of both relayed traffic as well as self-originated traffic. Note that a node can only transmit a maximum of S_{burst} data packets for every successful handshake, even if it has accumulated more than S_{burst} packets. Also, since there may be more than one neighbor for which the sender has accumulated at least S_{burst} packets for each of them, a simple heuristic is used to pick a receiver among them when this scenario arises:

1. Among the shortlisted candidates, a neighbor that is associated with the most accumulated relayed packets is picked as the target receiver, R_{target}. If there is more than one high priority buffer having this same number of packets, an R_{target} is chosen randomly among them. If there are S_{burst} or more relayed packets destined to R_{target}, the sender only tries to transmit S_{burst} packets. If the number of relayed packets is less than S_{burst}, those self-generated packets from the low priority buffer corresponding to R_{target} will also be sent, up to a combined total of S_{burst}.

2. If all high priority buffers are empty, the sender shall pick an R_{target} with the longest low priority queue, or randomly pick one should there be equally long queues, and transmit at most S_{burst} packets to it.

The "batch-by-time" mechanism demands a sender to initiate an RTS attempt if a time duration of T_{max} has already elapsed since the release from previous handshake, provided it has collected at least one packet. Here, a node only maintains a single global timer, instead of separate timers for different buffers. This helps to reduce packet delays when the load is low. Whenever the triggering condition is met, the sender uses a simple heuristic to pick its target receiver:

1. A sender inspects all its high priority buffers to identify the one that has the longest queue. The neighbor that is associated with this buffer is picked as R_{target}. If there is more than one high priority buffer having this same number of relayed packets, an R_{target} is chosen randomly among them. Those self-generated packets from the low priority buffer of R_{target} will also be sent.

2. If all high priority buffers are empty, the sender shall pick an R_{target} with the longest low priority queue, and break any tie randomly.

Besides satisfying either one of the above two triggering conditions, a node also must not have been required by some other nodes to remain silent, nor currently engaged in any other handshake. Otherwise, it must defer its RTS attempt until these constraints no longer hold.

As mentioned earlier, a simple BEB algorithm is adopted in the BiC-MAC protocol, since the investigation of an optimal backoff algorithm is beyond the scope of our study. In the BEB algorithm, each node doubles its backoff counter, B_{cnt}, in the event of an RTS failure, with an upper bound of $B_{\max}$; on the other hand, a node resets its backoff counter to a minimum value of $B_{\min}$, upon a successful handshake. The backoff interval, T_{bk}, of the node can be expressed as

$$T_{\text{bk}} = \mathit{uniform}\{0, B_{\text{cnt}}\} \times \tau_{\max}. \tag{4.10}$$

Upon satisfying either of the two RTS triggering conditions, an idle node initializes its contention timer based on T_{bk}, and only transmits an RTS upon its timer expiry.

4.3.3 Handling Problematic Scenarios in BiC-MAC

Although the idea of bidirectional transmissions may seem simple, one must pay careful attention in handling several problematic scenarios that may arise in underwater networks.

Scenario A: Fig. 4.2(a) shows a problematic scenario that is characterized by a transmission pattern in which two neighboring nodes, S_1 and S_2, transmit their RTS at around the same time. As can be seen, the bidirectional data transmissions are likely to cause a large number of consecutive data collisions at S_1 and S_2. Note that the data collisions caused by this transmission pattern are unique to the bidirectional transmission approach, as no data collision will occur if there were only unidirectional transmissions. Hence, we shall call it a "*Bidirectional Induced Data Collision (BIDC)*" problem.

To avoid the problem, both S_1 and S_2 shall revert to a unidirectional transmission approach upon detecting that Scenario A has occurred (see Fig. 4.2(b)). In particular, a sender must pay close attention to any overheard RTS packet (xRTS) during the WFCTS state, i.e., state in which it waits for the CTS from its intended receiver. This indicates that another sender is within its vicinity, and may potentially result in the BIDC problem. The sending of a data packet train instead of an NTF packet implicitly informs the receiver node that the bidirectional

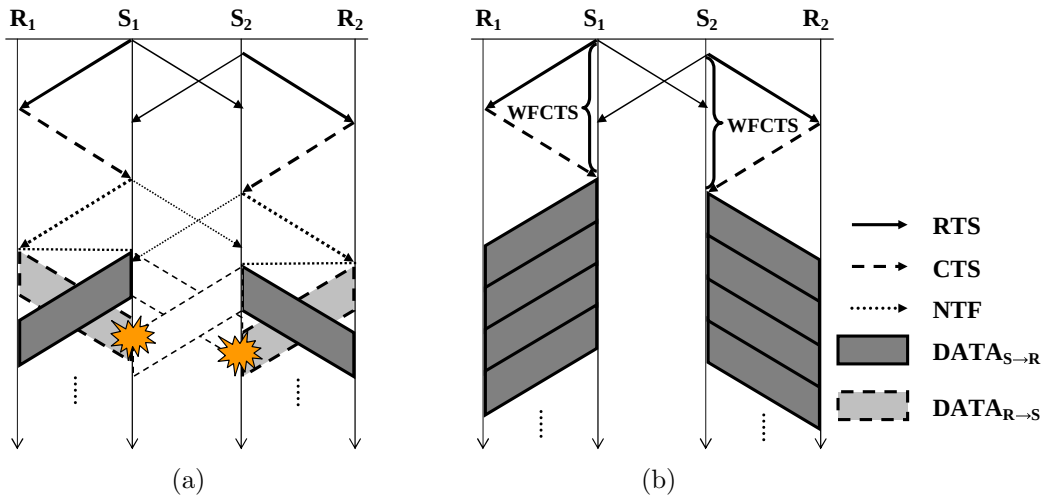


Figure 4.2: (a) Transmission pattern in Scenario A causes bidirectional induced data collision problem, (b) proposed solution for Scenario A.

transmission has been aborted. The neighbors of the intended receivers R_1 and R_2 may have longer silent durations than necessary, as R_1 and R_2 do not announce that packet exchange has become unidirectional transmissions. While this is conservative, the design is simple since they can continuously receive incoming unidirectional packets without having to explicitly transmit another control packet.

Scenario B: Fig. 4.3(a) shows a problematic scenario that is characterized by a transmission pattern in which two neighboring nodes, R_1 and R_2, receive their respective RTS at around the same time. A BIDC problem occurs in this scenario as well, where the packets transmitted by R_1 interfere with R_2's reception, and vice versa. Fig. 4.3(b) shows our proposed solution to avoid the problem. Here, R_1 and R_2 pay attention to any overheard CTS packets (xCTS) during the WFNTF state, i.e., the state in which a receiver node waits for the NTF packet. It indicates that a receiver is in its vicinity, and may cause the BIDC problem. In this case, R_1 and R_2 shall inform S_1 and S_2, respectively, by each transmitting an ABORT packet at its starting reference time t_{ref}. Upon receiving an ABORT rather than a data packet, the sender transmits its remaining data packets unidirectionally, without catering for the the bidirectional transmissions any further[3]. Note that it is also possible for the ABORT packet to collide with one or two data packets at the neighboring receiver node, which is a comparatively smaller loss.

Scenario C: Fig. 4.4(a) depicts a scenario in which R_1 and S_2 are within each other's communication range. As illustrated, a BIDC problem can also occur

[3]This is similar to [21], where a "WARNING" packet is used to defer unidirectional data transmission for avoiding collisions. However, upon receiving ABORT in our bidirectional case, the sender can still transmit unidirectionally in the current handshake.

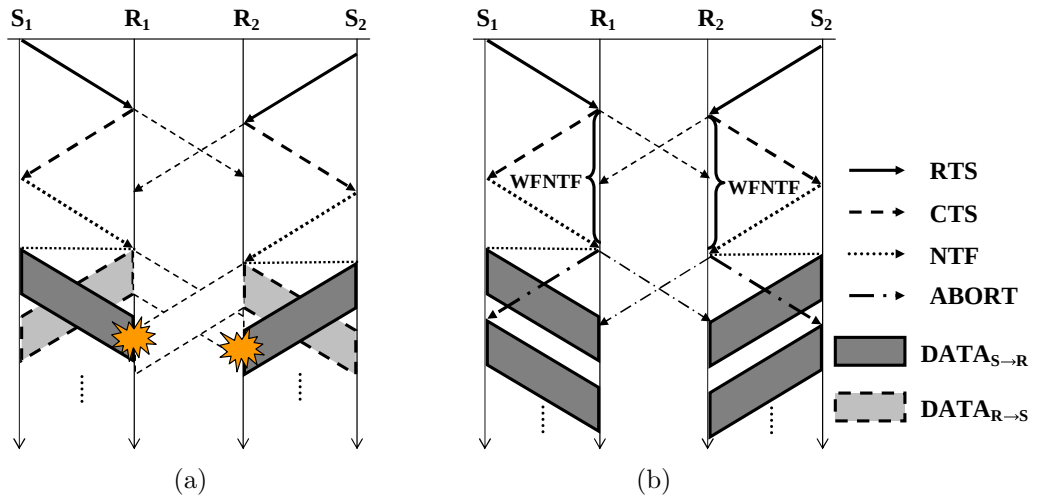


Figure 4.3: (a) Transmission pattern in Scenario B causes bidirectional induced data collision problem, (b) proposed solution for Scenario B.

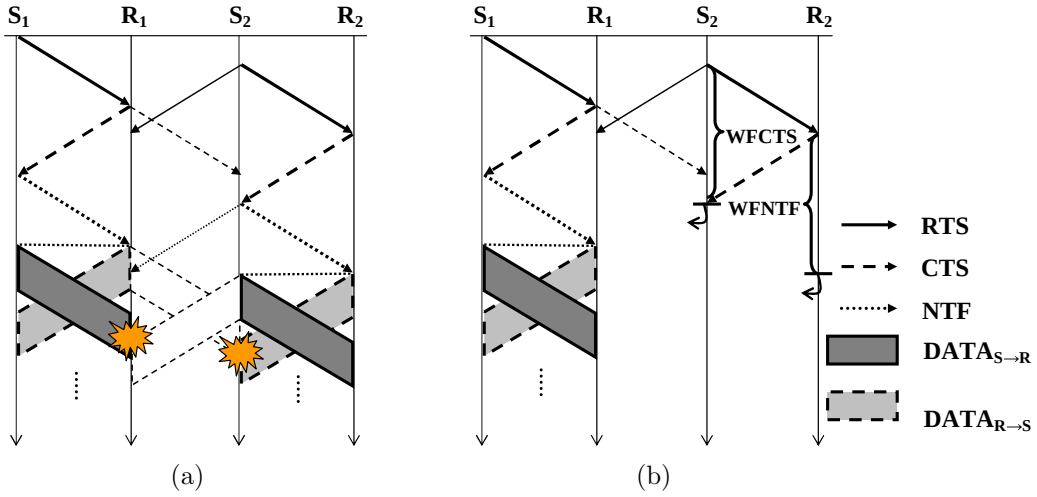


Figure 4.4: (a) Transmission pattern in Scenario C causes bidirectional induced data collision problem, (b) proposed solution for Scenario C.

in this scenario, where collisions occur at both R_1 and S_2, in contrast to only at R_1 for the unidirectional case. Unlike the solutions of Scenario A and B, here, one of the S-R node pairs can still proceed with the bidirectional packet transmissions (see Fig. 4.4(b)). In particular, after S_2 overhears xCTS from R_1 in the WFCTS state, it aborts its current handshake by ignoring the CTS from R_2. Subsequently, R_2 aborts its handshake when its WFNTF (wait for NTF) state timeouts. Therefore, only the S_1-R_1 node pair manages to perform the bidirectional transmissions.

Scenario D: Fig. 4.5(a) shows a scenario where a deadlock may occur when two neighboring senders repeatedly send RTS to each other, albeit the presence of a random backoff mechanism. Fortunately, this can be easily resolved as follows. Recall that in the RTS, a sender will announce the number of relayed packets and self-originated packets that it intends to transmit. Hence, a sender can figure out

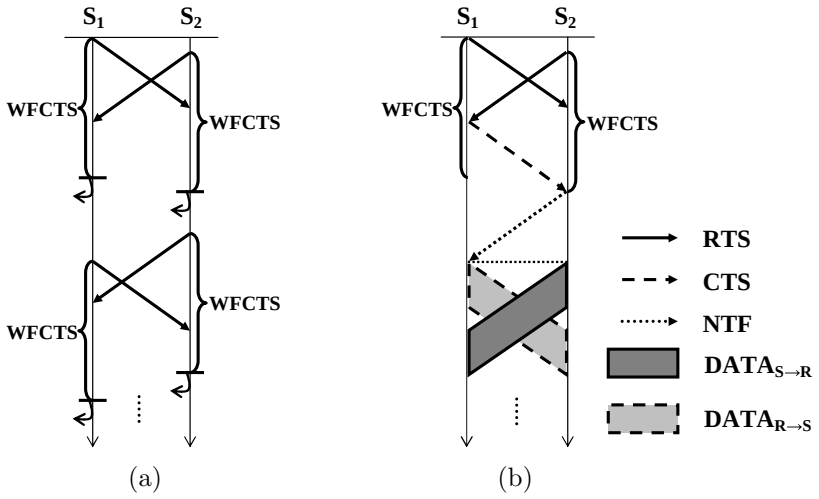


Figure 4.5: (a) Transmission pattern in Scenario D may result in a deadlock, (b) proposed solution for Scenario D.

its priority level against the other node according to a set of priority rules. A node first compares the total number of packets it intends to transmit with what the other node has indicated in its RTS. In the case of identical priority, it can further compare the total number of relayed packets. If both senders still have identical priority, then this can be resolved based on whose unique ID is larger. In Fig. 4.5(b), we assume that S_2 wins the priority test, and thus S_1 replies with a CTS. Note that both S_1 and S_2 can still exchange their packets bidirectionally.

4.3.4 Preventing Packet Drops at Relay Nodes

Here, we propose a packet drop prevention mechanism for BiC-MAC, which works well with both source-routing and hop-by-hop based routing schemes. Without such a mechanism, data packets are dropped at a relay node when its relay buffer is full, and this becomes more serious when operating at high load. Consequently, all valuable channel resources previously consumed are wasted. Note that for bidirectional transmissions, this problem may occur at both the initiating sender and its intended receiver. Table 4.2 defines some of the notation used to explain how our mechanism works.

For source-routing protocols (e.g., DSR [57]), each source node knows the exact path to all destinations from its route discovery algorithm, and this complete path is carried in every data packet's header. To prevent any packet drop, the RTS/CTS/NTF packets can be modified as follows. In the RTS packet, apart from specifying n_S, a sender declares how many packets are going to each of its

Table 4.2: Additional notation used in Section 4.3.4 and Section 4.3.5

Notation	Description
$n^{\text{intend}}_{x\to y\to z}$	Number of data packets that node x intends to transmit to its one-hop neighbor y for relaying to the latter's one-hop neighbor z
$n^{\text{actual}}_{x\to y\to z}$	Number of data packets that node x can transmit to its one-hop neighbor y for relaying to the latter's one-hop neighbor z, without causing any packet drop at node y
$Q^{\max}_x$	Maximum buffer size for packets of type x, where $x \in \{\text{relay}, \text{new}\}$
$Q_{x,i}(k)$	Average number of accumulated type x data packets destined to neighbor i for the k^{th}-interval, where $x \in \{\text{relay}, \text{new}\}$
$Q_i(k)$	Normalized k^{th}-interval queue occupancy for both relayed and new traffic destined to neighbor i
$I_{\text{ave}}(k)$	Average traffic intensity indicator at the k^{th}-interval
$\overline{I_{\text{ma}}}(k)$	Exponential moving average of traffic intensity indicator at the k^{th}-interval

intended receiver's next-hop neighbors, i.e., [next-hop i, $n^{\text{intend}}_{\text{S}\to\text{R}\to i}$]. Upon receiving the RTS, the receiver checks each of its corresponding relay buffers whether it can hold those incoming packets. Hence, the receiver can decide how many packets the initiating sender should transmit, and include its decision in the CTS reply, i.e., [next-hop i, $n^{\text{actual}}_{\text{S}\to\text{R}\to i}$]. A similar technique is used at the sender as well. The receiver that wishes to transmit n_{R} packets, can announce its packet composition in the CTS reply. The sender then decides how many packets the receiver should transmit, and includes its decision in its NTF response. We denote n'_{S} and n'_{R} as the total number of packets that the sender and the receiver should each transmit, respectively, without causing any packet drop. Note that a receiver must ensure that $n_{\text{R}} \leq n'_{\text{S}}$ when specifying n_{R} in its CTS reply, so that $n'_{\text{R}} \leq n'_{\text{S}}$ eventually.

For hop-by-hop routing protocols (e.g., AODV [58]), every node periodically updates its own routing table to all destinations, so that it knows exactly which next-hop neighbor should be chosen when it wishes to relay the incoming packets. Instead of announcing [next-hop i, number of packets] in the control packets, we can replace them with [destination i, number of packets]. Based on the destination ID, the intended receiver can look up its local routing table to figure out which next-hop neighbor should the packets be relayed to; thus, it can inspect the corresponding buffer's remaining spaces, and repeat the aforementioned procedures to prevent packet drops.

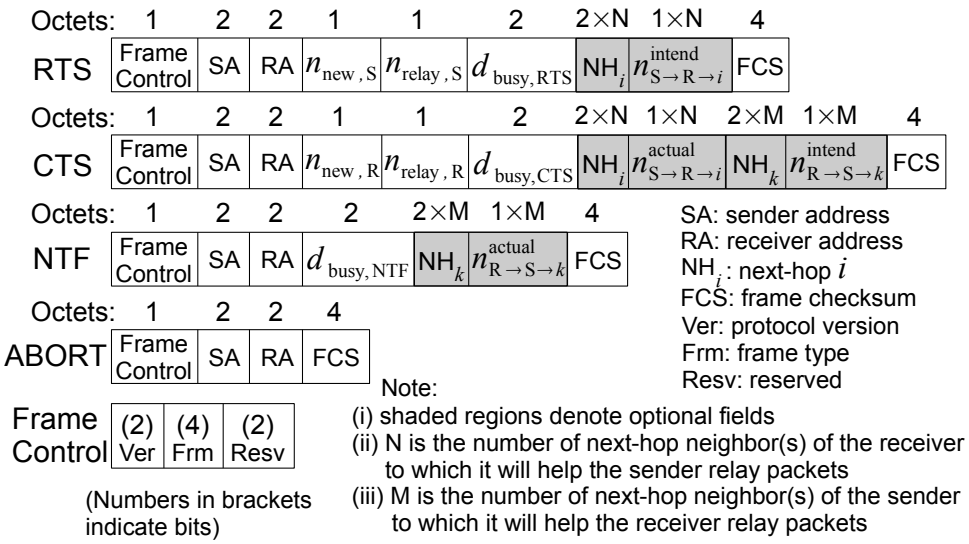


Figure 4.6: Control packet formats used in BiC-MAC. The shaded areas indicate the optional fields; they can be removed when the packet drop prevention mechanism is not utilized.

We now discuss how our modifications above affect the busy durations' computations, which will in turn influence the silent durations. Particularly, the computation of $d_{\text{busy,CTS}}$ in (4.1) is now over-conservative. When a receiver prepares its CTS, it has to assume that it can transmit n_{R} packets in its $d^{\text{rx}}_{\text{busy,R}}$ calculation, since the value of n'_{R} will only be known when the NTF packet arrives. Note that $n'_{\text{R}} \leq n_{\text{R}}$. Hence, the receiver's one-hop neighbors may set their silent durations to be longer than necessary. To overcome this problem, after n'_{R} is known, the receiver can announce the updated busy durations, which can be carried in its data packet's header in every CR. Then, all overhearing neighbors can update their silent durations accordingly. In contrast, the computation of $d_{\text{busy,NTF}}$ is not affected because the sender knows the exact n'_{S} and n'_{R} before it broadcasts the NTF. Note that our mechanism can also be employed in unidirectional handshaking-based protocols to prevent packet drops. Finally, we summarize our proposed control packet formats for BiC-MAC in Fig. 4.6.

4.3.5 Adaptive RTS Attempt Mechanism

So far, we have elaborated how S_{burst} and T_{max} work. In practice, however, it is difficult to determine the suitable values for these parameters because they depend on several factors such as the network topology, network traffic conditions, etc. More importantly, a dynamic scheme is desired because it is unlikely that a single set of static parameters would perform well across all network loads.

We propose a mechanism to dynamically adjust S_{burst} according to the run-time network loading measurements, while keeping $T_{\max}$ fixed. The network loading can be estimated using either: (i) queue occupancy measurements [59], or (ii) busy and idle period measurements [60]. Our main idea is to keep S_{burst} as small as possible when network load is low, and allow it to increase as network load grows. In this way, the "batch-by-size" mechanism becomes the primary triggering mechanism across all network loads. Similar to [56] and [61], $T_{\max}$ is set to a pre-defined constant. The reason we choose a constant $T_{\max}$ is explained as follows. In the adaptive mechanism, S_{burst} could occasionally be overestimated because the network load changes dynamically; this causes the delay performance to deteriorate due to the longer time taken to form the larger packet bursts. Hence, $T_{\max}$ can be used as a safeguard mechanism to ensure that the packet delay does not grow unrestrained. Its value may depend on the higher protocol layer; for instance, if TCP is utilized, $T_{\max}$ should not be greater than the value of retransmission timeout minus the round-trip time associated with a flow [56].

We explain the adaptive mechanism as follows. Table 4.2 defines some of the notation used in our explanation. As mentioned, each node maintains two separate buffers to store both relayed and self-originated packets for each of its N one-hop neighbors. In our mechanism, we partition time into intervals of equal duration. Each node keeps track of the average number of data packets for all its buffers (i.e., $Q_{x,i}(k)$, where $x \in \{\text{relay}, \text{new}\}, i \in \{1, \ldots, N\}$) in the k^{th}-interval, where $k \in \mathbb{Z}^+$ (i.e., positive integers). At the end of the k^{th}-interval, a node computes its normalized queue occupancy for each neighboring node i, as

$$Q_i(k) = \frac{Q_{\text{relay},i}(k) + Q_{\text{new},i}(k)}{Q_{\text{relay}}^{\max} + Q_{\text{new}}^{\max}},\ i \in \{1, \cdots, N\}, \tag{4.11}$$

where $Q_x^{\max},\ x \in \{\text{relay}, \text{new}\}$ denote the maximum buffer sizes for the respective types of packets. Then, the average traffic intensity indicator for k^{th}-interval is

$$I_{\text{ave}}(k) = \frac{\sum_{i=1}^{N} Q_i(k)}{N}. \tag{4.12}$$

In order to smooth out the random fluctuations, an exponential moving average is obtained as follows:

$$\overline{I_{\text{ma}}}(k) = \alpha \cdot \overline{I_{\text{ma}}}(k-1) + (1-\alpha) \cdot I_{\text{ave}}(k), \tag{4.13}$$

where α is a smoothing factor set to 0.9. We use this constant because previous work has shown that it is a good compromise between accuracy and promptness [62]. Nonetheless, we also study the sensitivity of the protocol's performance to α's value in Section 4.4. Note that $\overline{I_{\text{ma}}}(0) = 0$. Finally, the estimated S_{burst} for the next interval can be computed as

$$S_{\text{burst}}(k+1) = \left\lfloor S_{\text{burst}}^{\min} + \overline{I_{\text{ma}}}(k) \cdot [S_{\text{burst}}^{\max} - S_{\text{burst}}^{\min}] \right\rfloor , \tag{4.14}$$

where $S_{\text{burst}}^{\min}$ and $S_{\text{burst}}^{\max}$ are the respective minimum and maximum S_{burst} values. $S_{\text{burst}}^{\max}$ is a fixed value that is subject to buffer constraints, while $S_{\text{burst}}^{\min}$ is set to the smallest unit, i.e., $S_{\text{burst}}^{\min} = 1$. Whenever a node releases from a handshake, it uses the most recently estimated S_{burst} as a threshold to initiate its next RTS attempt. All nodes start with $S_{\text{burst}}(1) = 1$ and they adapt their own S_{burst} independently.

4.4 Performance of BiC-MAC in Multi-hop Networks

4.4.1 Simulation Model

We have developed a custom C++ discrete event-driven network simulator. As shown in Fig. 4.7, our multi-hop network topology has 36 static nodes with a grid spacing of 2000 m. The maximum communication range is 1.75 times the grid spacing, or 3500 m; the interference range is assumed to be the same as the communication range (note that in Section 4.6, we also study the effects of large interference range). Hence, each node has exactly 8 one-hop neighbors and 16 two-hop neighbors. A *wrap-around* strategy is used to distribute network load evenly and eliminate boundary effects. Each node generates its data packets according to the Poisson distribution, and randomly picks one of the 16 two-hop neighbors as a *destination*. Every node has a half-duplex omni-directional transceiver. The channel is also assumed to be error-free unless stated otherwise, so that packet losses are solely caused by packet collisions[4]. The acoustic propagation speed is 1500 m/s and the transmission rate is set to 4800 bps. Unless otherwise stated, the data packet length is 1200 bits, and the lengths of the RTS, CTS, NTF, and

[4]Similar to previous works [19, 20, 25–27, 34, 63], we mainly focus on the MAC protocol's behavior under the effects of long propagation delay and low bit-rate, and do not model packet loss due to channel effects of propagation loss model [64]. Nonetheless, we show the effects of packet error rate in Section 4.4.3.6.

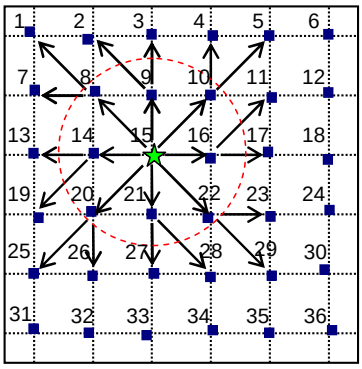


Figure 4.7: The multi-hop topology used in our simulations. Every other node assumes the same static routing pattern as the node marked with a star; we only consider two-hop routes so that it is easier to interpret the results. To resemble a real scenario, each node is allowed to randomly deviate from a grid intersection point by a maximum of 10% of the grid spacing, so as to introduce randomness.

ABORT packets are set to 152, 200, 136, and 72 bits, respectively. For the guard time and BEB algorithm, we set $T_{\text{guard}} = 10$ ms, $B_{\min} = 1$, and $B_{\max} = 32$. As for the packet train based protocols, our adaptive RTS attempt mechanism is also used; we set $S_{\text{burst}}^{\min} = 1$, $S_{\text{burst}}^{\max} = 200$, $\alpha = 0.9$, $S_{\text{burst}}(1) = 1$, $T_{\max} = 100$ s, and the updating interval is set to 50 s. In addition, all protocols are assumed to employ the packet drop prevention mechanism that we have introduced in Section 4.3.4. A node maintains two buffers (for relayed and new packets) for each of its one-hop neighbors, where each buffer can hold 100 packets. Lastly, we do not put any upper limit on the number of retries when RTS attempts fail. To avoid any transient effect, the simulation results are collected from 2×10^4 s to 1×10^5 s. Also, all results are averaged over 5 different topologies. Note that our topology covers a wide area of 100 km^2, similar to the practical Seaweb experiment [9]. Both transmission rate and communication range are referenced from the LinkQuest medium-range acoustic modem [65].

We use two variants of unidirectional handshaking-based MAC protocols to show the superiority of adopting a bidirectional, concurrent transmission approach in underwater networks.

- **MACA for Underwater (MACA-U) protocol**: In Chapter 3, we have proposed MACA-U as a reference MAC for handshaking-based protocols. Similar to MACA [13], its handshake consists of RTS/CTS/DATA. However, it has additional state transition rules to handle certain problematic scenarios that are likely to occur in underwater networks. All its configurable parameters follow the default settings as described above, except that each

node only maintains two buffers (one for relayed traffic and one for new traffic), with a size of 800 packets per buffer[5]. Its RTS and CTS packets' lengths are set to 104, and 88 bits, respectively.

- **MACA-U with Packet Train (MACA-UPT) protocol**: It is derived from MACA-U, except that a train of n_{S} data packets are sent unidirectionally for each successful handshake. Its RTS and CTS packets' lengths are set to 152, and 184 bits, respectively. Like BiC-MAC, MACA-UPT also maintains a dedicated pair of buffers for each of its one-hop neighbors. To facilitate an unbiased comparison, we adopt identical strategies as our BiC-MAC for the RTS attempts and backoff algorithm. Note that as there is no acknowledgement (ACK) packet involved, the sender's one-hop neighbors only need to remain silent for a short duration upon overhearing an xRTS to accommodate the subsequent CTS. Hence, these neighbors can participate in new handshakes sooner, which yields more parallel transmissions in the neighborhood. For comparison purpose, we also simulate a variant of MACA-UPT that requires the sender's one-hop neighbors to have longer silent durations upon overhearing an xRTS (their silent durations lasted till the end of the packet train's transmission), called "MACA-UPT-L".

4.4.2 Performance Metrics

We choose the following metrics to evaluate the MAC protocol's performance:

- **Normalized throughput per node (end-to-end)**[6], γ:

$$\gamma = \frac{1}{N} \cdot \frac{\sum_{i=1}^{i=N} r_i \cdot L_{\mathrm{DATA}}}{t_{\mathrm{sim}} \cdot R}, \tag{4.15}$$

where N is the total number of nodes, r_i is the total number of data packets successfully received by the *end destination* i, L_{DATA} is the packet's length in bits, t_{sim} is the simulation duration, and R is the link transmission rate (e.g., 4800 bps in our simulations).

[5]In our packet train based protocols, each node maintains two buffers of 100 packets per buffer for each of its first-hop neighbors, i.e., 8 neighbors in our topology. For a fair comparison, MACA-U's buffer size should be set to 800 packets per buffer so that the total buffer sizes per node remain the same.

[6]The normalization is necessary because, in a multi-hop setting, the system's *aggregate throughput* could grow with the network size due to concurrent transmissions in multiple regions that are sufficiently far apart.

- **End-to-end data packet delay**: It is defined as the duration from when a data packet arrives at a source's buffer to when it is successfully received at its end destination.

We will evaluate the above performance metrics with respect to the **normalized offered load per node**, defined as

$$\rho_{\text{node}} = \frac{\lambda_{\text{node}} \cdot L_{\text{DATA}}}{R}, \tag{4.16}$$

where λ_{node} is a node's average data packet arrival rate (in packets/s) at its MAC layer from its higher layer.

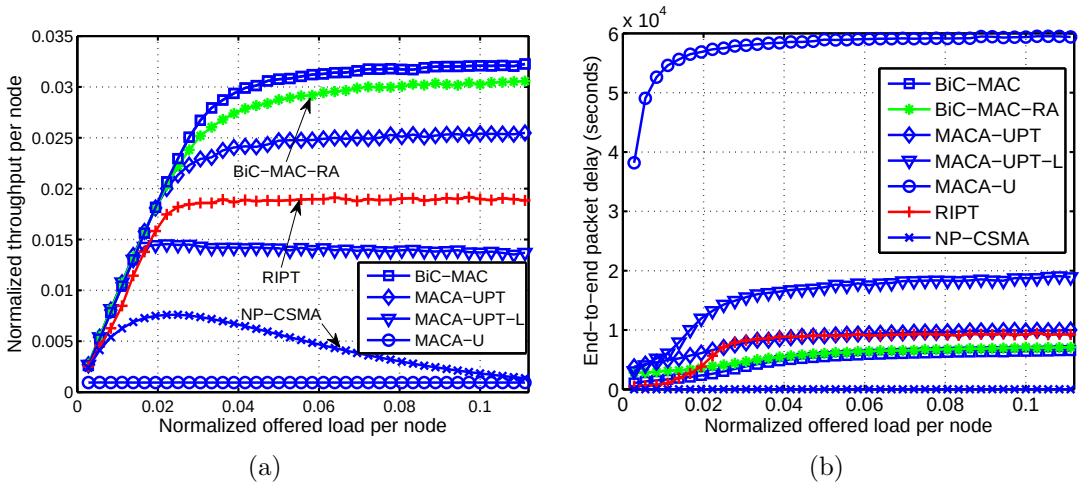
4.4.3 Simulation Results

To better understand the simulation results, we shall point out that the theoretical maximum (end-to-end) normalized throughput per node for our topology is only 0.1667. This can be explained as follows. By considering that the interference range is equal to the communication range, the number of possible concurrent transmissions without collision when the propagation delay is negligible is found to be at most 12 in our topology[7]. For the case where the propagation delay is large, we did not find any transmission pattern that outperforms the negligible propagation delay case. To find the maximum end-to-end performance for two-hop routes based on our topology, let us focus our attention on two arbitrary consecutive time slots. In the first time slot, we can have 12 source nodes concurrently transmit their "new" packets. Subsequently, in the second time slot, there will be at best 12 nodes that successfully relay those packets to their respective destinations. Thus, the maximum end-to-end normalized throughput per node is simply $1/2 \times 12/36 = 0.1667$. However, note that a practical MAC protocol may not achieve this performance as there are limiting factors such as idle backoff durations, packet collisions, control packet exchanges, and routing constraints.

4.4.3.1 Comparison against unidirectional-based MAC protocols

Fig. 4.8(a) shows the normalized throughput per node (hereafter referred to as "throughput") for several MAC protocols as we vary the normalized offered load

[7]Using Fig. 4.7, a possible set of 12 concurrent transmissions is demonstrated as follows: {[3,32],[4,11],[8,1],[9,16],[13,12],[14,21],[19,26],[24,17],[29,22],[30,31],[34,27],[35,6]}, where [x,y] denotes that node x transmits to node y.



(a) (b)

Figure 4.8: (a) Normalized throughput per node comparisons for various schemes, (b) end-to-end data packet delay comparisons for various schemes.

per node, while Fig. 4.8(b) illustrates their average end-to-end packet delay (hereafter referred to as "delay"). For comparison purpose, we also simulate a variant of BiC-MAC that relaxes the assumption of having the knowledge of first-hop neighbors, which we call "BiC-MAC-RA" (refer to Section 4.3.1.3 for its protocol implementation). Fig. 4.8(a)–4.8(b) show that BiC-MAC-RA not only significantly surpasses the rest of unidirectional-based protocols, but its performance is only slightly inferior to that of BiC-MAC; this suggests that our protocol can still perform well even in the absence of knowledge of the first-hop neighbors. Henceforth, we shall focus on BiC-MAC for the rest of the evaluations.

As illustrated, BiC-MAC and MACA-UPT variants outperform MACA-U in terms of throughput and delay due to the use of packet burst mechanism. As expected, MACA-U's throughput is unacceptably low, at around 0.001. In contrast, in the saturation throughput region, BiC-MAC greatly outperforms it by over 32.4 times. Clearly, it is costly to transmit only a single data packet for every handshake in a long propagation delay environment, since the proportion of time spent on the multi-way control packet exchange will become very significant. For the same reason, MACA-U has a very large delay as well.

Recall that our BiC-MAC protocol requires the first-hop neighbors of a sender to remain silent upon overhearing an xRTS packet until the end of the bidirectional packet exchange. While this requirement may appear inefficient, it is necessary for BiC-MAC to operate correctly. In contrast, MACA-UPT only requires these first-hop neighbors to remain silent for a short duration upon overhearing an xRTS packet. Thus, it benefits from better spatial reuse by

allowing these neighbors to participate in a new handshake sooner. By comparing the throughput of MACA-UPT with that of MACA-UPT-L, which intentionally extends the silent durations till the end of the packet train just to investigate the effect, we can see from Fig. 4.8(a) that the maximum throughput can drop by nearly 46.17% for the latter. This shows that, although BiC-MAC may be inefficient in terms of these long silent durations, its performance gains arising from the bidirectional concurrent transmissions are able to more than compensate for these inefficiencies. In the saturation throughput region, BiC-MAC surpasses MACA-UPT (MACA-UPT-L) by around 28.73% (135.23%). Furthermore, BiC-MAC has lower delay than MACA-UPT (MACA-UPT-L) across all offered load ranges; in particular, the saturation delay is reduced by around 35.84% (65.52%). Although BiC-MAC employs a 4-way handshake, as opposed to MACA-UPT's 3-way handshake, the performance gain has outweighed the overhead incurred by the use of an additional NTF packet. These aforementioned results confirm that a bidirectional transmission approach is indeed more efficient in the long propagation delay scenario, especially when operating at high load where the intended receiver can easily accumulate sufficient packets destined to the initiating sender. Without using this approach, the S-R node pair needs two sets of handshake to send data packets to each other, as opposed to BiC-MAC which requires only one.

We compare BiC-MAC against RIPT[8] [27], which is another handshaking-based MAC protocol that aims to offer high channel utilization in underwater networks. Note that although MACA-MN [25] also aims to achieve similar goal, we do not present its results because we found that it performs worse than RIPT[9]. Fig. 4.8(a)–4.8(b) show that BiC-MAC can offer remarkable improvements over RIPT; in particular, BiC-MAC achieves a saturation throughput gain of around 71.18%, as well as a saturation delay reduction of 29.41% over RIPT.

Finally, we simulate Non-Persistent Carrier Sense Multiple Access (NP-CSMA), where our implementation is similar to [63]. As shown, its throughput

[8]For RIPT protocol's parameters, we set $M_{\text{train,max}} = 120$ and $T_{\text{avg}} = 30$ s, where $M_{\text{train,max}}$ defines the maximum packet train size, while T_{avg} determines the average time between handshake initiations. The reason for $M_{\text{train,max}} = 120$ is because the BiC-MAC's adaptive train size will converge close to that value. In RIPT, each node only maintains two buffers (one for relayed traffic and one for new traffic), with a maximum size of 800 packets per buffer.

[9]Under *identical* multi-hop simulation conditions (i.e., topology, packet sizes, transmission link rate, traffic communication pattern), the RIPT protocol's simulation results in [27] have shown to be better than that of the MACA-MN's performance in [25].

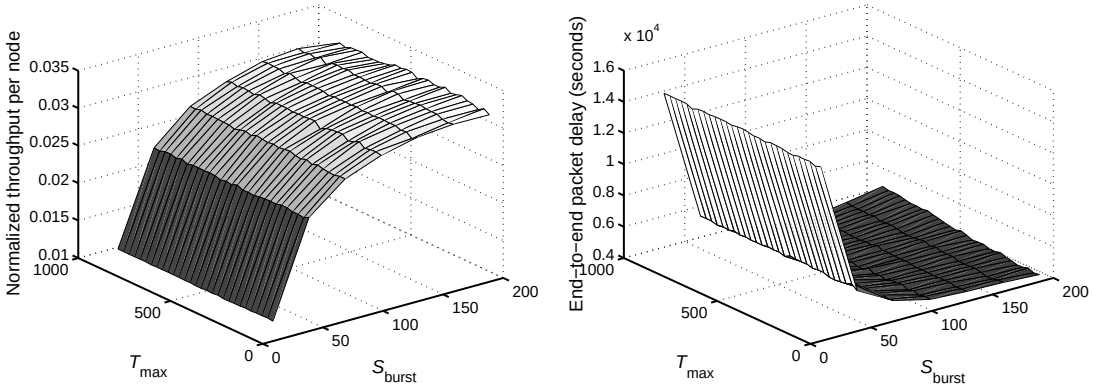
peaks at around 0.007 when normalized offered load per node is 0.022; beyond this, its throughput drops as NP-CSMA does not use collision avoidance scheme in the presence of hidden nodes. The performance of handshaking-based MACA-U is sensitive to inter-nodal distances [66]. While it performs poorly compared to NP-CSMA, especially in our current case of large grid size, we found that NP-CSMA suffers from significantly higher collision rates, compared to the rest of handshaking-based MAC protocols.

4.4.3.2 Effects of varying S_{burst} and T_{max}

We study the effects of varying S_{burst} and T_{max} on the BiC-MAC protocol for two operating load regimes, namely, normalized offered load per node of: (i) 0.0056 (low load), and (ii) 0.0417 (high load). S_{burst} and T_{max} are varied from 10 to 190, and 10 to 850 s, respectively.

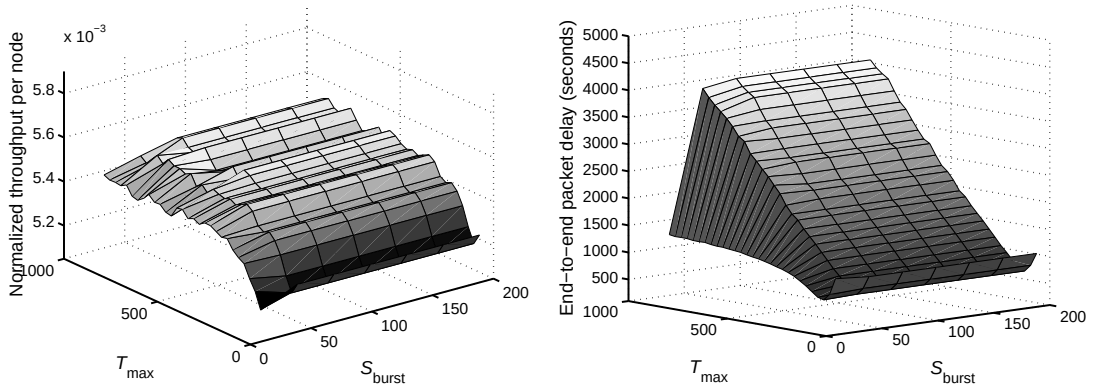
As shown in Fig. 4.9(a), when BiC-MAC operates at high load, its throughput increases rapidly as S_{burst} grows, but it eventually stabilizes. Similarly, its delay decreases rapidly as S_{burst} grows, but it eventually stabilizes as well (Fig. 4.9(b)). This is explained as follows. As S_{burst} gradually increases, a node can transmit more packets for each successful handshake so that the proportion of time spent on exchanging control packets becomes less significant; thus, both throughput and delay improve accordingly. However, as S_{burst} grows further, this effect diminishes; this is because it becomes harder to meet the triggering condition of "batch-by-size", and thus RTS attempts are more and more often triggered by the "batch-by-time" mechanism. Therefore, T_{max} must be carefully chosen as it affects the protocol's performance when "batch-by-size" mechanism loses its effects beyond a certain large S_{burst} value.

As illustrated in Fig. 4.10(a), when BiC-MAC operates at low load, its throughput and delay remain almost consistent across all S_{burst} range. This is because for the low packet arrival rate, it is very hard to meet the "batch-by-size" triggering condition, especially when S_{burst} is set to a large value; therefore, the RTS attempts are mainly triggered by the "batch-by-time" mechanism, and varying S_{burst} has little effect on the performance. Fig. 4.10(b) shows that there is an optimal point for which T_{max} can minimize the packet delay. When T_{max} is too small, a node triggers its RTS attempts very often, and the resulting intense contentions will cause more CTS failures; thus, more retransmissions



(a) (b)

Figure 4.9: Effects of varying S_{burst} and T_{max} on the BiC-MAC protocol's: (a) normalized throughput per node when normalized offered load per node is set to 0.0417, (b) end-to-end data packet delay when normalized offered load per node is set to 0.0417.



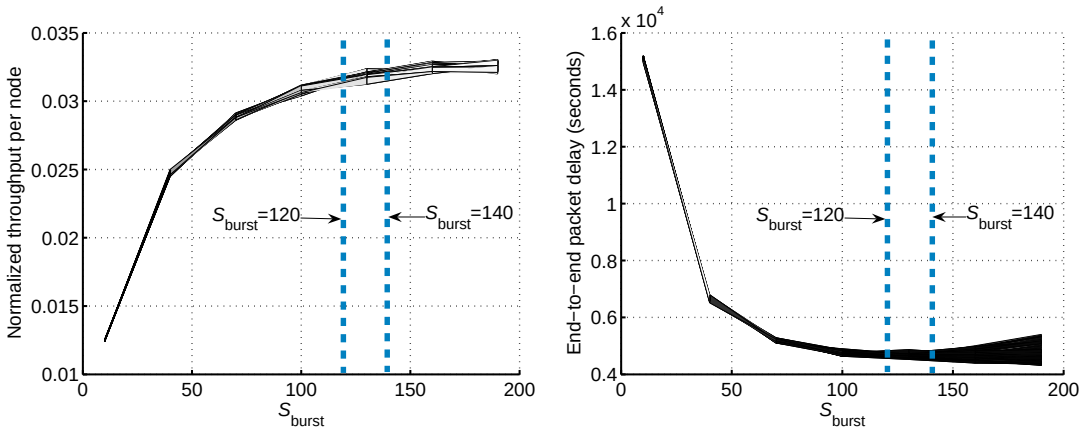
(a) (b)

Figure 4.10: Effects of varying S_{burst} and T_{max} on the BiC-MAC protocol's: (a) normalized throughput per node when normalized offered load per node is set to 0.0056, (b) end-to-end data packet delay when normalized offered load per node is set to 0.0056.

are required and larger delays are expected. Moreover, a node does not have enough opportunity to accumulate more packets before its RTS triggering, and this causes the communication overhead for each successful handshake to become more significant. On the other hand, when T_{max} is too large, the inter-RTS triggering time becomes larger than necessary, and thus the delay becomes larger as well.

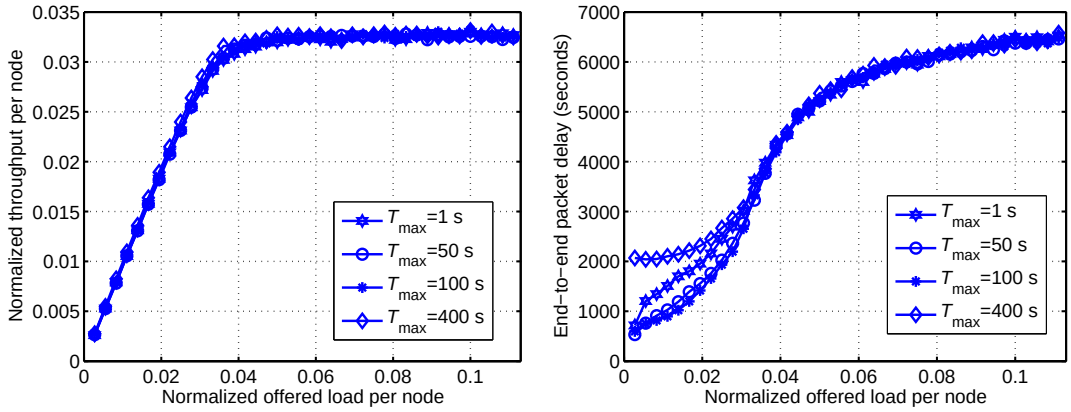
4.4.3.3 The optimal S_{burst} and T_{max} settings

We shall identify the optimal S_{burst} and T_{max} settings from Fig. 4.9(a)–4.9(b) and Fig. 4.10(a)–4.10(b), because they will be used to benchmark the effectiveness of our adaptive RTS attempt mechanism.



(a) (b)

Figure 4.11: (a) 2-D plot of Fig. 4.9(a) to show the BiC-MAC's normalized throughput per node versus S_{burst}, (b) 2-D plot of Fig. 4.9(b) to show the BiC-MAC's end-to-end data packet delay versus S_{burst}.

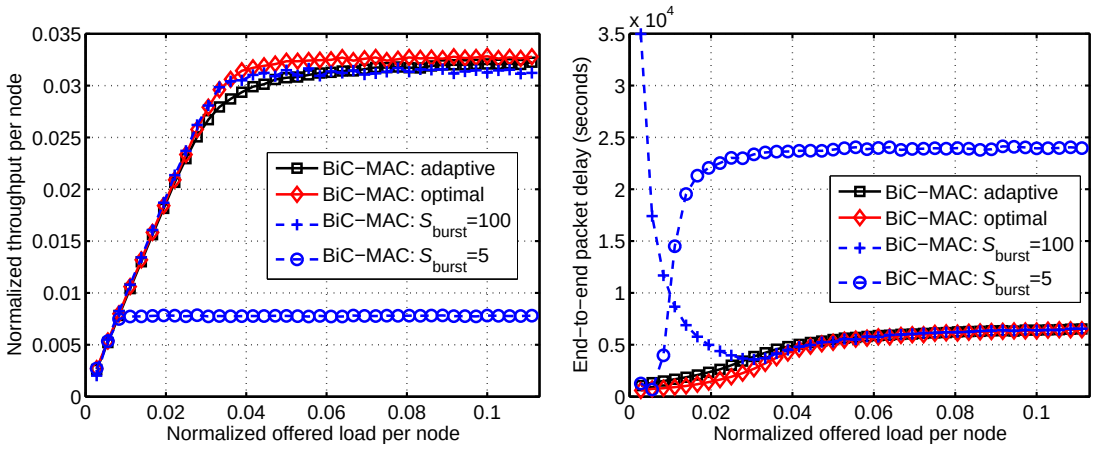


(a) (b)

Figure 4.12: (a) Effects of varying $T_{\max}$ on the BiC-MAC's normalized throughput per node when $S_{\text{burst}} = 130$, (b) effects of varying $T_{\max}$ on the BiC-MAC's end-to-end data packet delay when $S_{\text{burst}} = 130$.

We first explain how S_{burst} can be chosen. Fig. 4.11(a) and Fig. 4.11(b) show the corresponding 2-D plots of BiC-MAC's throughput and delay, respectively, when it operates at high load (Fig. 4.9(a)–4.9(b)). As shown, a suitable S_{burst} range could be $[120, 140]$, in which throughput and delay are kept within a satisfactory level. Recall that the "batch-by-time" mechanism will become the main triggering mechanism when S_{burst} is set to a very large value. In this case, if we also set $T_{\max}$ to a large value, then a larger delay is resulted as the inter-RTS triggering time now mainly depends on $T_{\max}$. Thus, we should not consider $S_{\text{burst}} > 140$ due to the large delay variations across $T_{\max}$ range. $S_{\text{burst}} < 120$ is also not suitable because both throughput and delay will deteriorate.

We now discuss how $T_{\max}$ can be chosen. Fig. 4.12(a) and Fig. 4.12(b) show



(a) (b)

Figure 4.13: Performance comparisons of BiC-MAC that utilizes the adaptive RTS attempt mechanism against several other schemes: (a) normalized throughput per node, (b) end-to-end data packet delay.

the effects of different $T_{\max}$ on BiC-MAC's throughput and delay, respectively, when S_{burst} is fixed at 130. As shown, varying $T_{\max}$ only affects the delay at low load, and does not significantly impact both throughput and delay at high load. At low load, a node needs relatively longer time to accumulate the required S_{burst} packet, thus making the "batch-by-time" mechanism mainly responsible for triggering the RTS attempts. In contrast, the "batch-by-size" is the main triggering mechanism at high load, and the varying of $T_{\max}$ has little effect. Therefore, we utilize BiC-MAC's delay performance at low load to determine an appropriate $T_{\max}$ range. A suitable range is found to be within $[70, 130]$ s because BiC-MAC could maintain a low packet delay when operating in that region.

4.4.3.4 Performance of the adaptive RTS attempts mechanism

We now examine the performance of BiC-MAC when the adaptive RTS attempt mechanism is utilized. Fig. 4.13(a) and Fig. 4.13(b) show the performance comparisons in terms of throughput and delay, respectively, with several other variants that do not utilize such a mechanism.

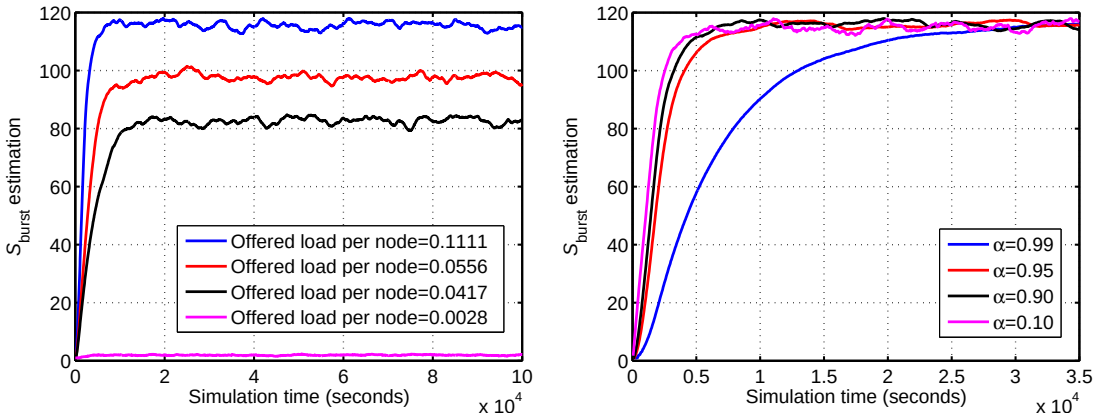
We first explain why an adaptive mechanism is necessary by demonstrating that the non-adaptive variants do not work well across all offered loads. For the non-adaptive variants, where S_{burst} is set to 5 or 100, $T_{\max}$ is assumed to be infinity so that all RTS attempts are triggered by the "batch-by-size" mechanism. As shown, BiC-MAC with $S_{\text{burst}} = 5$ performs well at low load, but its performance deteriorates quickly as the load increases. This is because a smaller S_{burst} yields a larger communication overhead as the sender only transmits a short packet burst

for every handshake. In contrast, BiC-MAC with $S_{\text{burst}} = 100$ performs better at high load, but its packet delays are very high at low load. In short, regardless of T_{max} settings, S_{burst} should be kept as small as possible at low load, and allowed to increase as the load grows. We now compare the adaptive BiC-MAC against an optimal BiC-MAC, in which its S_{burst} and T_{max} are set respectively to 130 and 100 s, as discussed previously. We observe that the adaptive BiC-MAC's performance is very close to that of the optimal scheme. A key advantage of an adaptive mechanism is that it can tune its parameters based on real-time loading, as opposed to using simulations to obtain the optimal settings.

Finally, we also study two important aspects of the adaptive mechanism: (i) convergence of S_{burst}, and (ii) protocol's sensitivity to the smoothing factor α. Fig. 4.14(a) shows the convergence performance for different offered loads. As shown, the estimated S_{burst} values converge and they have already stabilized when the results are collected from 2×10^4 s. Within a few hundred seconds, the adaptive mechanism is able to estimate reasonably good S_{burst} values ($S_{\text{burst}} > 80$), which can offer normalized throughput per node of around 0.03, as shown in Fig. 4.11(a). Fig. 4.14(b) illustrates the effects of varying α, when the normalized offered load per node is set to 0.1111. As shown, the convergence time remain quite short for α up to 0.95; further increasing α to 0.99 makes the transient phase longer, but its curve becomes smoother. In contrast, when $\alpha = 0.1$, the transient phase is shortest, but there are more fluctuations in S_{burst}. Nonetheless, we observe that the variations in S_{burst} is not significant. Thus, the protocol's performance is not highly sensitive to α so long as it is not too large. In BiC-MAC, we choose $\alpha = 0.9$ because it appears to be a good compromise between promptness and stability.

4.4.3.5 Effects of varying inter-nodal distances

Table 4.3 illustrates the effects of varying inter-nodal distances (grid sizes) on BiC-MAC's saturation throughput and delay. In general, underwater handshaking-based MAC protocols are very sensitive to the inter-nodal propagation delay, as a multi-way handshake is needed prior to the data packet transmissions. As the inter-nodal distance increases, the communication overhead is amplified, which leads to performance degradation. However, as can be seen, for a larger inter-nodal distance, both BiC-MAC's throughput gain and delay improvement over the unidirectional MACA-UPT become more significant.



(a) (b)

Figure 4.14: (a) Convergence of S_{burst} estimation for different normalized offered load per node, (b) effects of varying the smoothing factor α on S_{burst} estimation when normalized offered load per node is set to 0.1111.

Table 4.3: Saturation Throughput Per Node and End-to-End Packet Delay Comparisons for Different Inter-nodal Distances

Grid	BiC-MAC		MACA-UPT		Throughput	Delay
Sizes	Throughput	Delay	Throughput	Delay	Gain	Reduction
1000 m	0.0329	5867 s	0.0311	7857 s	5.79%	25.33%
3000 m	0.0300	7288 s	0.0220	11835 s	36.36%	38.42%
5000 m	0.0259	8778 s	0.0168	15602 s	54.17%	43.74%
7000 m	0.0230	10203 s	0.0140	18321 s	64.29%	44.31%
9000 m	0.0206	11521 s	0.0119	21303 s	73.11%	45.92%

4.4.3.6 Effects of varying packet error rate

We now explore the effects of imperfect channel, in which both control and data packets are subjected to certain packet error rate (PER). Our goal is to study the protocol's robustness, since underwater channel is unreliable [1]. Our simulation model is presented as follows. Whenever a node is listening to a packet, the received data packet of length L_{DATA} is subjected to a pre-defined PER, say ψ_{DATA}; while the control packet of length L_x, where $x \in \{\text{RTS}, \text{CTS}, \text{NTF}, \text{ABORT}\}$, will be subjected to a PER of $\psi_x = (L_x/L_{\text{DATA}}) \cdot \psi_{\text{DATA}}$. Fig. 4.15 shows the effects of varying PER on BiC-MAC's saturation throughput, when the normalized offered load per node is fixed at 0.1111. As shown, the throughputs of all schemes degrade gracefully without any sudden drops. The BiC-MAC protocol also outperforms the rest of the protocols across all simulated PER ranges, offering a throughput of around 0.018 when $\psi_{\text{DATA}} = 0.3$.

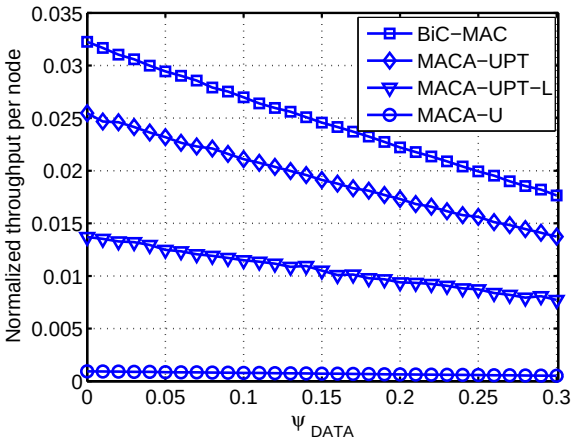
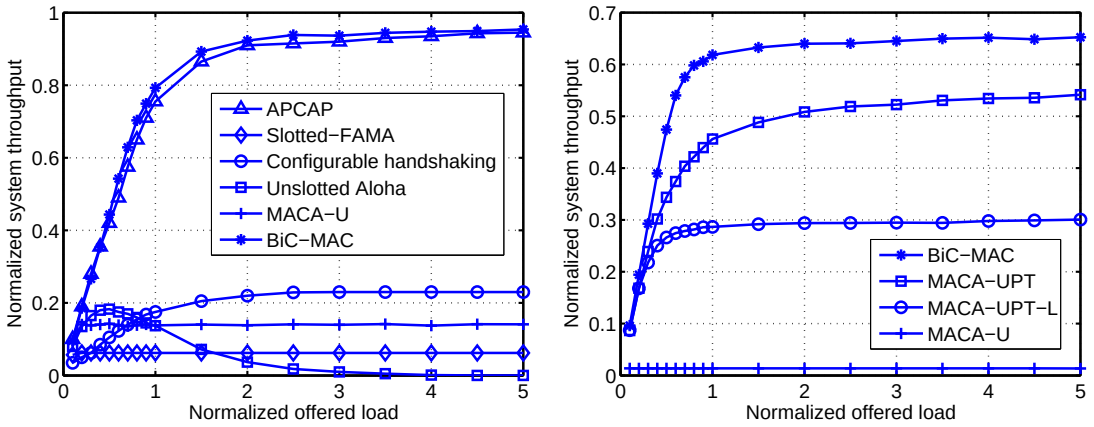


Figure 4.15: Effects of varying packet error rate on the BiC-MAC protocol's normalized saturation throughput per node. Here, the normalized offered load per node is fixed at 0.1111.

4.5 Performance of BiC-MAC in Single-hop Networks

4.5.1 Simulation Model

We compare BiC-MAC against several other schemes in single-hop networks. Guo *et al.* [20] have compared their proposed APCAP protocol with several selected MAC protocols such as the Slotted-FAMA [19], configurable handshaking protocol [21], and unslotted-Aloha. We follow the same simulation setup. Specifically, 20 static nodes are randomly deployed in a 4500×4500 m^2 area, transmission rate is fixed at 2400 bps, and data packet length is set to 8000 bits. In [20], their RTS and CTS packets' lengths are 160 and 176 bits, respectively. In contrast, the lengths of our RTS, CTS, NTF, and ABORT are set to 104, 104, 88, and 72 bits, respectively. Note that these packet lengths are shorter than those in the multi-hop settings because we do not need to include information for preventing packet drop at relay nodes in the case of single-hop networks. The data packet generated is allowed to be destined to any one of the neighbors with equal probability. Here, the adaptive RTS mechanism is still employed, in which $S_{\text{burst}}^{\min} = 1$, $S_{\text{burst}}^{\max} = 100$, $\alpha = 0.9$, $S_{\text{burst}}(1) = 1$, $T_{\max} = 100$ s, and the updating interval is 50 s. For the packet train based MAC protocols, each node maintains a single buffer of 100 packets for each of its neighbors; while MACA-U only maintains a single buffer with a size of 1900 packets. All results are averaged over 5 different topologies.



(a) (b)

Figure 4.16: (a) Normalized system throughput comparisons for BiC-MAC against several other selected MAC protocols in [20], (b) normalized system throughput comparisons for BiC-MAC against our proposed reference MAC protocols, when the data packet length is set to 600 bits.

4.5.2 Simulation Results

Fig. 4.16(a) shows that BiC-MAC achieves the highest system throughput, and it performs marginally better than APCAP across all offered load ranges. However, most of its transmissions had reverted to unidirectional transmissions because the transmission duration of the larger packet size used in [20] simply does not allow for bidirectional transmissions. Nevertheless, we also simulate a scenario to study the bidirectional gain over its unidirectional counterparts by reducing the data packet length to 600 bits while keeping all other settings the same. As can be seen in Fig. 4.16(b), BiC-MAC has a saturation throughput gain of around 21% over the unidirectional MACA-UPT; this shows that the bidirectional transmission approach is still useful in single-hop networks.

For the rest of the MAC protocols, we see that the configurable handshaking protocol outperforms Slotted-FAMA. Although Slotted-FAMA uses a time-slotting mechanism to avoid data collisions, its long slot size requirement will seriously degrade its throughput. In contrast, the configurable handshaking protocol utilizes a more efficient handshake, in which a sender is allowed to use different handshake lengths for different receivers so that the average handshake duration can be minimized. As expected, the unslotted-Aloha cannot maintain its throughput as the load grows because it does not employ any collision avoidance mechanism. Interestingly, MACA-U performs better than Slotted-FAMA for all offered loads, as well as the configurable handshaking protocol in the low load region, even though MACA-U only transmits a single data packet for each successful handshake.

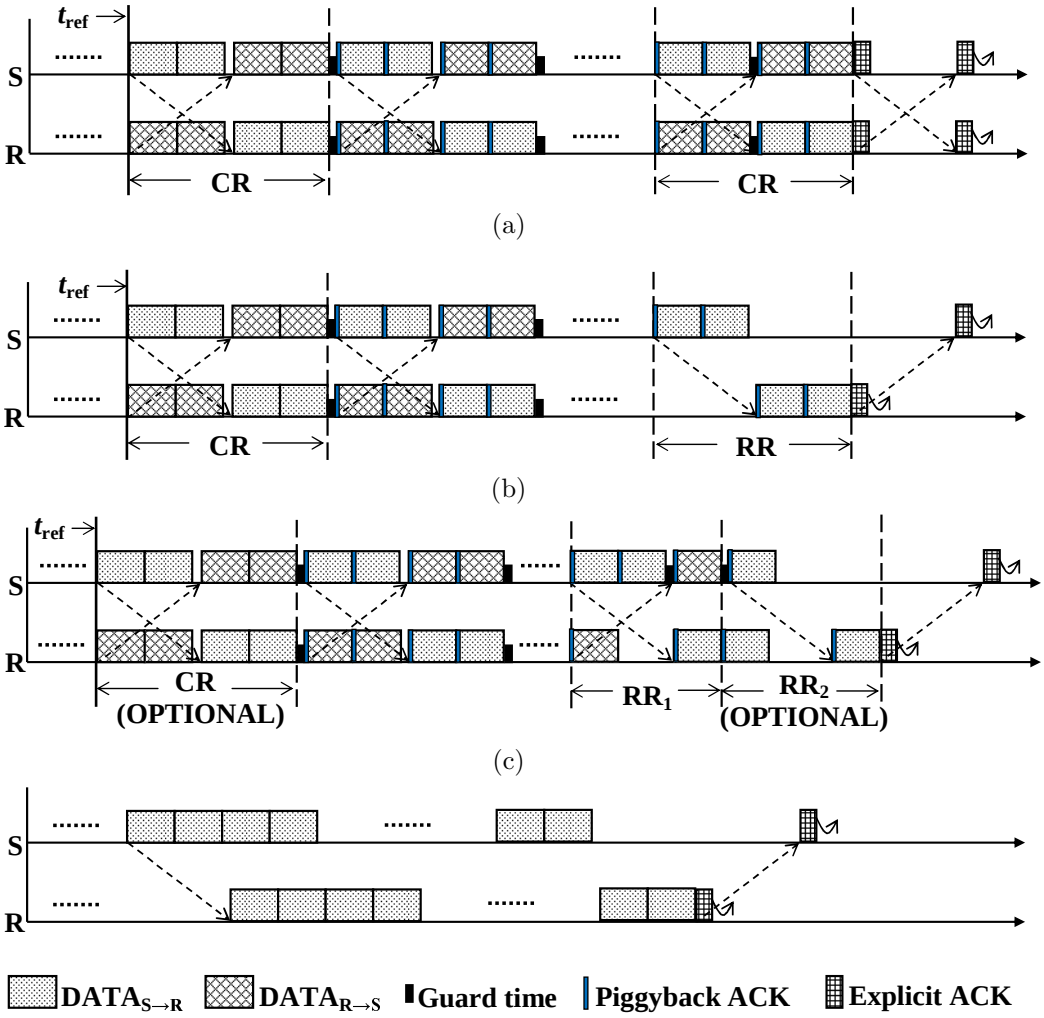
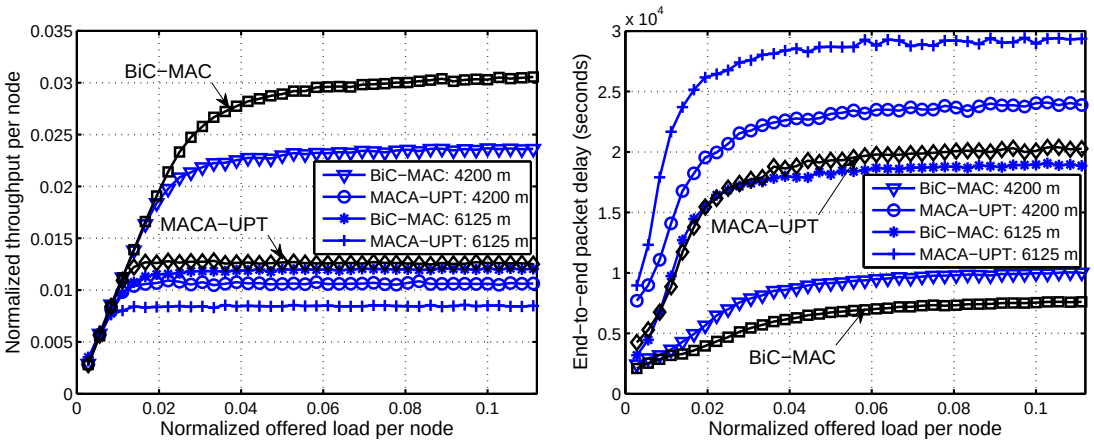


Figure 4.17: BiC-MAC with ACK enhancement: (a) Type 1, (b) Type 2, (c) Type 3 scenarios. In every round, ACKs are piggybacked in each transmitted data packet to indicate erroneous received packets from the previous round. A stand-alone, explicit ACK is also used to inform the receiving status of last round. Figure (d) shows that only a single explicit ACK is employed in the unidirectional transmission scenario (e.g., MACA-UPT).

4.6 Discussion

Finally, we discuss two important issues: incorporation of Automatic Repeat reQuest (ARQ) error control technique, and the effects of large interference range on our protocols. We intend to enhance BiC-MAC with ARQ mechanism, so as to allow retransmission of any unsuccessfully received packets. By using the inherent bidirectional transmission framework in BiC-MAC, we propose a new "*Bidirectional Burst ACKnowledgement (Bi-BACK)*" mechanism. Fig. 4.17(a)–4.17(c) show the timing diagrams of Bi-BACK for the three bidirectional scenarios in BiC-MAC.

In general, every transmitted data packet has error control bits like cyclic redundancy check (CRC), which can be used at a receiving node for packet error detection [29]. As shown, starting from the second CR, both nodes in S-R node pair can insert ACKs in each transmitted packet to announce only those erroneous received packets from the immediately preceding CR (i.e., negative ACKs [29]). Similarly in RR, ACKs are carried in each data packet, but sent unidirectionally. To improve robustness, redundant ACKs can be piggybacked in every packet in each transmission round. A stand-alone, explicit ACK is also used to feedback the receiving status of last reception round. Note that in the BiC-MAC's Type 3 scenario with the configurations of: either CR and RR_1, or RR_1 only, two explicit ACKs need to be exchanged bidirectionally (not shown in Fig. 4.17(c) for brevity). We point out that when channel is very bad, a transmitted packet could be totally undetected at a receiving node. In this case, we cannot rely on the packet's CRC for error detection. In BiC-MAC, however, each node in S-R pair knows how many packets to be exchanged in every round. Thus, even though a packet cannot be successfully detected, a node is still able to conclude that its received packet is lost and that packet will be negative-acknowledged. When a sender discovers any unsuccessful packet receptions at the receiver, all data packets that are negatively acknowledged can be retransmitted along with any accumulated relayed and self-originated traffics, in its next handshake loop. To reduce packet delay, we prioritize those packets that are waiting for retransmission over relayed and self-originated traffics. Note that a node still uses "batch-by-size" and "batch-by-time" mechanisms in Section 4.3.2 to determine its RTS attempt triggering. For the unidirectional data transmission, an intended receiver only transmits a single explicit ACK to announce its packet receiving status, as shown in Fig. 4.17(d). To avoid interfering with explicit ACK receptions, overhearing nodes should extend their silent durations accordingly (i.e., $d^{\text{CTS}}_{\text{silent},i}$ and $d^{\text{NTF}}_{\text{silent},i}$). Note, however, that the proportion of silent duration extension for the bidirectional transmission scenarios, is much lesser than that of the unidirectional case. For example, upon overhearing an xRTS, the first-hop neighbors of an initiating sender in MACA-UPT, must now extend their silent durations to accommodate for ACK in return (for the case without using ACK, those neighbors only need much shorter silent duration for CTS in return); in contrast, in our previous BiC-MAC without ACK, the first-hop neighbors were required to wait until the end of bidirectional packet exchange.



(a) (b)

Figure 4.18: Comparing the effects of using ACK mechanism in BiC-MAC and MACA-UPT, as well as the impacts of large interference range on these MAC protocols with ACKs: (a) normalized throughput per node, (b) end-to-end data packet delay.

Hence, upon introducing ACK mechanism, we expect MACA-UPT to show a more evident performance degradation.

Compared to the conventional Stop-and-Wait ARQ variants in [29], which only send a single unidirectional ACK at the end of packet receptions, our bidirectional ACK approach has two advantages. First, Bi-BACK is more reliable and robust, since our MAC framework allows multiple ACKs to be feedbacked periodically, and not subjected to a single ACK packet loss. Second, our scheme greatly improves retransmission efficiency in the event of ACK packet loss. This is unlike the unidirectional ACK case, in which *all* previously sent data packets (even if received correctly) need to be retransmitted, because the sending node is uncertain about the packet receiving status of each data packet at the receiver.

We now show the effects of employing ACKs in BiC-MAC and compare its performance against MACA-UPT with ACK, in terms of normalized throughput per node and end-to-end packet delay (labeled as "BiC-MAC" and "MACA-UPT" in Fig. 4.18(a)–4.18(b)). Unless otherwise stated, we follow the same settings of 36-node multi-hop simulation in Section 4.4.1. For unidirectional transmissions, the length of an explicit ACK is 328 bits; for bidirectional transmissions, the lengths of a piggyback ACK and explicit ACK are set to 48 and 136 bits, respectively. We do not put any limit on the number of retransmissions when data packet is not received correctly. For now, interference range is assumed to be the same as the communication range of 3500 m. With reference to BiC-MAC and MACA-UPT

without ACKs in Fig. 4.8(a)–4.8(b), we can see that the addition of ACK causes much significant deterioration of throughput and delay in MACA-UPT, compared to BiC-MAC. This can be attributed to two factors. First, MACA-UPT with ACK now requires the first-hop neighbors of a sender to extend their silent durations to account for ACK in return, and thus cannot benefit from spatial reuse (similar to the earlier "MACA-UPT-L"). Second, MACA-UPT needs to retransmit the entire block of packets, if the feedbacked ACK packet is unsuccessfully received at the sender node. Lastly, we explore the effects when interference range is larger than communication range[10]. In the simulation, the interference range is set to 1.20 [67] or 1.75 times the communication range of 3500 m (i.e., 4200 m or 6125 m). Note that for our multi-hop grid topology, the latter range is large enough to cause packet interference to all 16 second-hop neighbors. When interference range is set to 4200 m or 6125 m, the saturation throughput for BiC-MAC with ACK drops by around 22.7% or 60.6%; its corresponding saturation delay also degrades by 31.8% or 148.0%. As interference range grows, it greatly limits BiC-MAC performance, since increasingly more number of second-hop neighbors in an S-R pair fail to receive packets correctly. Nonetheless, we still found that BiC-MAC achieves better throughput and delay performance than MACA-UPT with ACK, under both simulated interference ranges.

4.7 Summary

In this chapter, by exploiting the acoustic channel's unique characteristics, we have presented a novel approach based on bidirectional, concurrent data packet exchange to improve the data transmission efficiency in underwater acoustic networks. The opportunity of bidirectional packet exchange is only possible in high latency networks, and should occur in a concurrent manner so that packet transmissions are tightly packed. To further amortize the high latency communication overhead, we also present a packet bursting idea, where an S-R node pair can exchange multiple rounds of bidirectional packet transmissions.

Based on the above strategies, we propose a single-channel, asynchronous handshaking-based MAC protocol, called BiC-MAC, that aims to offer high channel utilization. Although the handshake is sender-triggered, an intended receiver

[10]Similar to [34] that evaluates MAC performance in grid topology, we also focus on a fixed interference range model.

can exploit that opportunity to initiate multiple rounds of bidirectional, concurrent data exchange if it has packets in return. Unlike most proposed MAC protocols that adopt unidirectional data exchange, both sender and its intended receiver in BiC-MAC, share only one set of communication overhead and thus the proportion of time spent on control signaling is greatly reduced. A normal unidirectional data transmission can still be performed, if the receiver does not have any packets in return. Thus, MACA protocol with packet train can be regarded as a special case of BiC-MAC. Furthermore, BiC-MAC is designed with a versatile MAC framework to support all possible modes of bidirectional transmissions, so as to cater for the fact that S-R node pair may not intend to exchange the same number of packets. We also identify and resolve four major problematic scenarios in BiC-MAC, such as the BIDC problem that arises due to the use of bidirectional transmissions.

From our extensive single-hop and multi-hop simulations, as well as comparisons with several other existing MAC protocols such as MACA-U, MACA-UPT, RIPT, slotted FAMA, etc., we demonstrate that BiC-MAC can significantly outperform the conventional unidirectional handshaking-based protocols in terms of both throughput and delay, while still offering a stable saturation throughput. We also show that the BiC-MAC variant that relaxes the assumption of having first-hop neighbors' knowledge, is only slightly inferior than the optimized BiC-MAC, and can still achieve remarkable performance improvement compared to the existing unidirectional-based protocols. In addition, we show that the adaptive RTS attempt mechanism in BiC-MAC, which tunes S_{burst} parameter based on run-time network loading, can offer a performance very close to that of the optimal scheme. Under error-prone channels, BiC-MAC also surpasses other unidirectional-based protocols and its throughput degrades gracefully without any sudden drops. Our study highlights the value of performing bidirectional-concurrent data exchange in a long propagation delay environment. Based on our findings, we contend that BiC-MAC is an ideal candidate for networks with limited mobility. Moreover, it is also particularly suited for delay-tolerant applications that constantly operate at high load, such as oceanic data collection, undersea exploration, and etc.

Having investigated the BiC-MAC's performance through simulations, we will analytically study the single-hop saturation throughput performance of a time slotted BiC-MAC variant in Chapter 6.

Chapter 5

A MAC Protocol with Reverse Opportunistic Packet Appending

5.1 Introduction

In Chapter 4, we have shown that BiC-MAC can remarkably improve channel utilization, via exploiting long propagation delay to perform bidirectional-concurrent data exchange. However, this bidirectional exchange approach is not feasible when the inter-nodal propagation delay is smaller than the transmission time of a single data packet, which may occur in short and medium-range underwater networks. In this case, its data exchange procedure will revert to a conventional unidirectional approach. The bidirectional transmission approach also yields little gain when an intended receiver node only has a few or no data packets in return. Furthermore, BiC-MAC limits the packet exchange to happen only between a sender and its intended receiver in each handshake. None of the potentially backlogged neighbors of the sender can take advantage of the current handshake to transmit its own packets. Each of them must initiate its own handshake, which incurs higher overheads.

In addition to the BiC-MAC's shortcoming, the existing handshaking-based MAC protocols described in Chapter 2, are often plagued with two limitations. First, some MAC protocols [9,10,18–21] only allow an initiating sender to transmit a single data packet for each successful handshake. When coupled with the long delays incurred during the multi-way control packet exchange (exacerbated in long propagation delay environment), this results in *large signaling overhead*. Second, in order to reduce the large signaling overhead, several other MAC protocols

adopt a packet train approach, where a sender transmits a block of data packets to a single receiver [24, 26], or to multiple receivers [25]. However, their data transmission phase is still quite inefficient, since only a single node is allowed to transmit in each handshake. More importantly, this single-node transmission approach does not exploit the opportunity of concurrent transmissions, which is inherent in underwater acoustic networks. Therefore, this often leads to *inefficient channel utilization*. Finally, a recently proposed MAC protocol, called RIPT [27], uses receiver-initiated reservations to allow multiple nodes to transmit their packet trains in each handshake. It has demonstrated that a multi-node transmission approach could be beneficial. However, the channel utilization of RIPT is still quite low, mainly because of its inefficient 4-way handshake mechanism.

In this chapter, a new approach is proposed to address the above problems. Our idea exploits two unique opportunities that arise from the use of handshaking-based protocols in long propagation delay settings. First, to enhance channel utilization, one could "reuse" the 2-way RTS/CTS handshake by better utilizing the long idle waiting time associated with multi-way control packet exchange in high latency scenario (handshake's round-trip delay). One possible way is that in each handshake, an initiating sender can take that opportunity to poll and setup multiple backlogged neighbors for subsequent data transmissions. This would be more efficient than most conventional handshaking-based MACs, which only allow packet transmissions that originate from a single sender (i.e., single-node transmission approach). Second, unlike terrestrial wireless networks, concurrent packet transmissions in a long propagation delay environment do not always result in packet collisions. Hence, multiple sender nodes can be scheduled to transmit with partial overlap in transmission time, as we only need to ensure that no overlapping occurs at the intended receiver.

Based on these observations, we propose a novel MAC protocol based on reverse opportunistic packet appending (ROPA). ROPA is a single-channel, sender-initiated MAC protocol that does not require any clock synchronization. Similar to some of the aforementioned protocols, it seeks to improve channel utilization by reducing the proportion of time spent on control signaling. However, it achieves this differently by allowing an initiating sender to invite its first-hop neighbors (appenders) to opportunistically transmit (append) their data packets to the initiating sender. Specifically, after the initiating sender finishes transmitting

its data packets to its receiver (primary data transmissions in the *forward path*), it can start to receive the incoming appended data packets during the same handshake from multiple neighbors (secondary data transmissions in the *reverse path*). This is in contrast to the conventional approach, which requires each of those neighbors to initiate a separate handshake that incurs its own overheads. Also, in order to achieve better channel utilization, those appended packets are scheduled to arrive in a collision-free, packet train manner even though they may consist of multiple blocks of data packets sent from different appenders. While this may require some appenders to transmit with partial overlap in time due to different inter-nodal propagation delays, there is no collision so long as no packet overlapping occurs by the time the data packets arrive at the initiating sender.

Similar to other handshaking-based MAC protocols, ROPA uses information extracted from the control packets to alleviate the hidden node problem. ROPA also has a versatile MAC framework; when none of the sender's neighbors has any packet to append, it can still perform its forward path's transmissions, and this reduces to MACA [13] with packet train. Conversely, if the sender only receives its neighbors' appending requests, but does not hear from its own intended receiver, it can still receive the reverse path's packets, and this reduces to RIPT for the current handshake. Therefore, both MACA with packet train and RIPT can be regarded as special cases of ROPA.

The remainder of this chapter is organized as follows. We first describe our system model in Section 5.2. In Section 5.3, we explain the protocol design in detail. Next, we present the simulation results of ROPA for both multi-hop and single-hop networks in Section 5.4 and Section 5.5, respectively. We also discuss several important issues in Section 5.6. Finally, a conclusion is given in Section 5.7.

5.2 System Model

We consider an ad hoc, static, multi-hop acoustic network, in which each node is equipped with an omni-directional, half-duplex underwater acoustic modem. Note that these "static" nodes may still exhibit some limited movement as they are typically anchored to the seabed, and subjected to a maximum sway movement caused by underwater currents. As for our MAC protocol, both control and data packets are exchanged in a single channel. Currently, we only consider fixed-length

packets. We do not put any upper limit on the number of retries when RTS attempts fail. It is worthwhile to note that ROPA does not need clock synchronization. We also assume that every node knows its estimated propagation delay from each of its first-hop neighbors, and the estimated inter-nodal propagation delays between its first-hop and second-hop neighbors. During network initialization, this information is estimated via round-trip time (RTT) measurements of control packets [26] and some information sharing among neighboring nodes. Any estimation errors and the fluctuations caused by sway movements are typically very small compared to inter-nodal propagation delays; hence, their effects could be easily accommodated with the use of guard times.

5.3 The ROPA Protocol

5.3.1 Design Philosophy

We first discuss our design philosophy behind the packet appending idea.

5.3.1.1 Motivations for Appending in the Reverse Path

We offer some insights on why packet appending should be performed in the *reverse path*, as opposed to the forward path. First, the time spent for the polling/request procedures of the packet appending is shorter, as it fully overlaps with the RTS/CTS handshake duration. In particular, an RTS plays the additional role of polling whether the neighbors (including the intended receiver) have any data packets to append to the initiating sender. This is more efficient than if we were to append packets to the receiver node (i.e., forward path appending), due to the fact that the receiver could only poll its neighbors via its CTS packet, which will result in longer handshake negotiation time. Second, the reverse packet appending approach is more intuitive for a multi-hop packet forwarding scenario. Specifically, after the initiating sender emptied a portion of its buffer by transmitting its primary data packets, the incoming appended packets can then utilize this buffer space.

5.3.1.2 Caveat in Assigning Secondary Data Transmissions

In each handshake, the negotiation of packet appending can be accomplished via multi-way polling/request/grant procedures between an initiating sender "S"

and its first-hop neighbors. Upon collecting all appending requests, the sender needs to determine *when* and *how many* packets a potential appender should append, before broadcasting its decision (grant). When assigning secondary data transmissions, the sender "S" should ensure that: (i) the appended packets from different appenders arrive at the sender "S" in a collision-free manner, such that the latter can continuously receive those packets, without any idle gap (except for a small guard time between packet bursts from different appenders), and (ii) the appended packet transmissions must also not interfere with the primary packet receptions at the intended receiver "R", especially for those appenders that are within communication range with node "R". In our scheduling algorithm (explained later in Section 5.3.3.2), the sender cycles through each of the requests by first computing a valid packet appending timing for a given request entry, before allocating secondary data slots. Using the knowledge of estimated inter-nodal propagation delays, the sender could determine proper transmission timing, subjected to the above two constraints. Whenever a current request entry does not meet the constraints, the sender shall attend to subsequent set of requests, hoping that different potential appenders might satisfy the requirements, before revisiting this entry again.

5.3.1.3 Dealing with Packet Interference at Neighboring Nodes of an Appender

To allow better channel utilization, each granted appender needs not reserve its floor for the entire duration of its secondary data transmissions. Its neighboring nodes are then free to participate in another handshake. In this case, however, the appended packet transmissions may potentially interfere with packet receptions at its neighboring nodes. While the appended packet transmission timing computed by an initiating sender "S" seeks to provide collision-free packet receptions at both sender "S" and its intended receiver "R", this schedule does not avoid any potential packet interference at the neighboring nodes of an appender. This is due to the sender "S" lacks of sufficient knowledge regarding the channel activities around an appender. As detailed in Section 5.3.5, in order to alleviate excessive data collisions, each appender should not just blindly follow the sender's decision. Instead, it could actively listen to all control packets (i.e., indicating a nearby receiving node) before its packet appending, and will abort its scheduled appending,

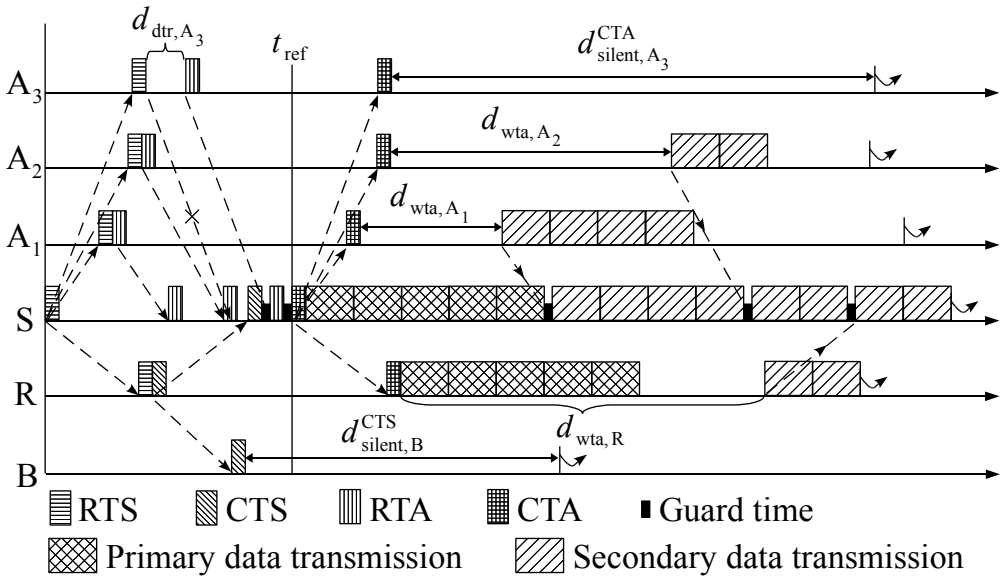


Figure 5.1: Timing diagram of the ROPA protocol. Nodes "S" and "R" refer to the initiating sender and its intended receiver, respectively. Nodes "A_1", "A_2", and "A_3" are first-hop neighbors of node "S", and can potentially become the appenders for this handshake. Node "B" is a first-hop neighbor of node "R" only, and is hidden from node "S". Note that small guard times can be inserted to account for any slight estimation errors of the inter-nodal propagation delays. The instances at which the nodes are released from the current handshake are shown as curly arrows.

if necessary. Although the initiating sender will waste some reserved data slots, it can still receive appended packets from other granted appenders.

5.3.2 How the ROPA Protocol Works

We now explain the working mechanism of ROPA. Fig. 5.1 illustrates the ROPA protocol's timing diagram, while Table 5.1 defines the notation used in our explanation.

5.3.2.1 Channel Reservation Phase

In ROPA, the RTS packet serves two purposes. First, it informs the receiver about the request to perform primary data transmissions, so that the receiver will try to reserve the floor around its neighborhood if it can indeed accept the packets. Second, it is used for polling the potential appenders whether they have any data packets to append. If a neighbor has packets to append, it requests for permission by using a Request-to-Append (RTA) packet. As shown in Fig. 5.1, the RTA packets might result in RTA-RTA or RTA-CTS collisions at the sender if

Table 5.1: Notation Used for Explaining the ROPA Protocol

Notation	Description
T_{DATA}	Transmission time of each fixed-length data packet
T_x	Transmission time of each fixed-length control packet of type x, where $x \in \{\text{RTS}, \text{CTS}, \text{RTA}, \text{CTA}\}$
$\tau_{i,j}$	Propagation delay between node i and node j
$\tau_i(k)$	Propagation delay between node i and an appender that has been assigned the order k, where $i \in \{\text{S}, \text{R}\}$
$T_{\max}$	Time threshold for triggering an RTS attempt
S_{pri}	Maximum allowable number of primary data packets that can be sent in a single handshake. It also serves as a threshold number of accumulated data packets for triggering an RTS attempt
S_{sec}	Maximum allowable number of secondary data packets that can be appended in a single handshake
T_{guard}	Guard time to accommodate any error in the inter-nodal propagation delay's estimation
$d_{\text{dtr},i}$	Defer-to-Request; duration that node i must wait before starting to transmit its RTA packet to avoid collision
$d_{\text{wta},i}$	Wait-to-Append; duration that node i must wait before starting to transmit its secondary data packets
$d^x_{\text{silent},i}$	Duration that node i must remain silent after overhearing a type x control packet, where $x \in \{\text{RTS}, \text{CTS}, \text{RTA}, \text{CTA}\}$
$d_{\text{busy},x}$	Busy duration information that is included in a type x control packet, where $i \in \{\text{RTS}, \text{CTS}, \text{RTA}, \text{CTA}\}$
$d_{\text{rel,S}}$	Duration that a sender spends from t_{ref} till it releases from the current handshake
$t^{\text{RX}}_{\text{RTS}}(k)$	Time instant at which an appender of order k finishes overhearing the RTS packet
$t^{\text{TX}}_{\text{RTA}}(k)$	Time instant at which an appender of order k starts transmitting its RTA packet
t_{ref}	Time instant at which sender starts sending CTA packet
$n^{\text{pri}}_{x,\text{S}}$	Number of primary data packets of type x that an initiating sender "S" intends to transmit in current handshake, where $x \in \{\text{relay}, \text{new}\}$
$n^{\text{req}}_{x,i}$	Number of data packets of type x that node i intends to append in current handshake, where $x \in \{\text{relay}, \text{new}\}$
$n^{\text{grant}}_{x,i}$	Number of data packets of type x that node i is allowed to append in current handshake, where $x \in \{\text{relay}, \text{new}\}$
$\mathcal{N}_i$	Set of node i's first-hop neighboring nodes

each potential appender were to simply respond with its RTA immediately. This can happen when two or more potential appenders have similar distances from the sender (e.g., nodes "A_2" and "A_3"). We solve this problem by requiring the sender to compute whether each neighboring node needs to delay its RTA response (details in Section 5.3.3.1), and include the schedule in its RTS. The sender specifies four types of information in the RTS: the total number of primary data packets it wishes to transmit, $n_{\mathrm{S}}^{\mathrm{pri}}$[1]; the maximum allowable number of secondary data packets that can be appended for the current handshake, S_{sec}; its expected busy duration, $d_{\mathrm{busy,RTS}}$; and the collision-free RTA schedule it has computed for its neighbors. The busy duration will be used by all overhearing nodes to compute their silent durations. For the collision-free schedule, the RTS packet only needs to explicitly specify the defer-to-request duration of an appender node i, $d_{\mathrm{dtr},i}$, if it is not allowed to transmit its RTA packet immediately after receiving the RTS. This limits the amount of information that needs to be carried in the RTS packet.

Upon overhearing an RTS packet, each potential appender i checks whether it has any data packets in return. If it does not, it extracts the RTS's busy duration to locally compute its silent duration, $d_{\mathrm{silent},i}^{\mathrm{RTS}}$; if it does, it prepares its RTA packet with the 3-tuple, $[n_{\mathrm{relay},i}^{\mathrm{req}}, n_{\mathrm{new},i}^{\mathrm{req}}, d_{\mathrm{busy,RTA}}]$, where $n_{\mathrm{relay},i}^{\mathrm{req}}$ and $n_{\mathrm{new},i}^{\mathrm{req}}$ are the number of relayed and new packets that it wishes to append, respectively; $n_{\mathrm{relay},i}^{\mathrm{req}} + n_{\mathrm{new},i}^{\mathrm{req}} \leq S_{\mathrm{sec}}$. The expected busy duration declared in the RTA packet, $d_{\mathrm{busy,RTA}}$, is the time needed before the node is expected to receive the sender's Clear-to-Append (CTA) packet, which in turn carries the decision on whose packet appending requests have been granted. The purpose of $d_{\mathrm{busy,RTA}}$ is to allow other neighbors that overhear the RTA to compute their respective silent durations, so they can avoid interfering with CTA reception. Before responding with an RTA, each potential appender must look up the collision-free schedule given by the RTS packet. When the schedule does not require the potential appender to defer its RTA transmission, it can do so immediately. Otherwise, it defers its RTA transmission by the amount of time indicated within the schedule.

For the intended receiver "R", upon receiving an RTS, it checks whether it has any data packets in return before responding with a CTS. Unlike the RTA response, the receiver can transmit its CTS without any deferment. When the receiver wishes to append packets, it specifies $n_{\mathrm{relay,R}}^{\mathrm{req}}$ and $n_{\mathrm{new,R}}^{\mathrm{req}}$ in its CTS reply;

[1]In multi-hop networks, each node generates its own data packets, and also helps its neighbors relay their packets. Thus, $n_{\mathrm{S}}^{\mathrm{pri}} = n_{\mathrm{relay,S}}^{\mathrm{pri}} + n_{\mathrm{new,S}}^{\mathrm{pri}}$.

$n^{\text{req}}_{\text{relay,R}} + n^{\text{req}}_{\text{new,R}} \leq S_{\text{sec}}$. When the receiver does not have any packet to append, the $n^{\text{req}}_{\text{relay,R}}$ and $n^{\text{req}}_{\text{new,R}}$ fields are set to zero. Regardless of the case, the CTS always carries a busy duration field, $d_{\text{busy,CTS}}$. Note that a potential appender (including receiver "R") can only transmit an RTA (CTS) packet if it is currently not involved in any other handshake and no other node requires it to remain silent.

Unlike the original MACA protocol where an initiating sender waits unproductively for the CTS, ROPA's sender can utilize this time window to collect some of the RTA requests, and collect any remaining RTA requests after receiving the CTS. After broadcasting an RTS packet, a sender waits until a reference time, t_{ref}, that is sufficiently large to accommodate even its most distant first-hop neighbor's incoming RTA packet. Having acquired all incoming requests, the sender allocates its available secondary data slots, S_{sec}, by using a simple strategy that prioritizes all relayed packets over new packets (to be detailed in Section 5.3.3.2). Then, a CTA packet is broadcasted to inform those granted appenders when and how many packets they should append. For each of the granted appenders, the CTA carries the following 4-tuple, [node ID, $n^{\text{grant}}_{\text{relay},i}$, $n^{\text{grant}}_{\text{new},i}$, $d_{\text{wta},i}$], where $n^{\text{grant}}_{\text{relay},i}$ and $n^{\text{grant}}_{\text{new},i}$ are the number of relayed and new packets that appender node i can transmit, respectively, and $d_{\text{wta},i}$ is the duration that node i must wait before it can start to append its packets. Similar to the other control packets, the CTA also contains a busy duration field, $d_{\text{busy,CTA}}$. Immediately after transmitting the CTA, the sender starts transmitting its primary packets to its intended receiver.

As mentioned, every control packet carries a busy duration field, which will be used by all overhearing nodes to calculate their respective silent durations. They are computed as follows:

$$\begin{cases} d_{\text{busy,RTS}} = t_{\text{ref}} - T_{\text{RTS}}, \\ d_{\text{busy,CTS}} = 2\tau_{\text{S,R}} + T_{\text{CTA}} + n^{\text{pri}}_{\text{S}} T_{\text{DATA}} \\ \qquad\qquad + [t_{\text{ref}} - (2\tau_{\text{S,R}} + T_{\text{RTS}} + T_{\text{CTS}})], \\ d_{\text{busy,RTA}} = 2\tau_{\text{S},i} + T_{\text{CTA}} \\ \qquad\qquad + [t_{\text{ref}} - (2\tau_{\text{S},i} + T_{\text{RTS}} + d_{\text{dtr},i} + T_{\text{RTA}})], \\ d_{\text{busy,CTA}} = d_{\text{rel,S}} - T_{\text{CTA}}, \end{cases} \tag{5.1}$$

where $i \in \mathcal{N}_{\text{S}} \setminus \{\text{R}\}$, and $d_{\text{rel,S}}$ is the duration that an initiating sender spends from t_{ref} till it releases from the current handshake ($d_{\text{rel,S}}$ can be computed using

(5.7)). The respective silent durations are:

$$\begin{cases} d^{\mathrm{RTS}}_{\mathrm{silent},i} = d_{\mathrm{busy,RTS}} + T_{\mathrm{CTA}}, & i \in \mathcal{N}_{\mathrm{S}}, \\ d^{\mathrm{CTS}}_{\mathrm{silent},j} = d_{\mathrm{busy,CTS}} - 2\tau_{\mathrm{R},j}, & j \in \mathcal{N}_{\mathrm{R}}, \\ d^{\mathrm{RTA}}_{\mathrm{silent},k} = d_{\mathrm{busy,RTA}} - 2\tau_{i,k}, & i \in \mathcal{N}_{\mathrm{S}} \setminus \{\mathrm{R}\}, k \in \mathcal{N}_i, \\ d^{\mathrm{CTA}}_{\mathrm{silent},i} = d_{\mathrm{busy,CTA}} - 2\tau_{\mathrm{S},i}, & i \in \mathcal{N}_{\mathrm{S}}. \end{cases} \tag{5.2}$$

Notice that, the silent duration (except $d^{\mathrm{RTS}}_{\mathrm{silent},i}$) computed by an overhearing neighbor can be shorter than the busy duration contained within the control packet; this is because of inter-nodal propagation delays. As for $d^{\mathrm{RTS}}_{\mathrm{silent},i}$, it should be sufficiently large so that all potential appenders can fully receive the CTA packet, which contains vital information for the forthcoming data packet appending.

5.3.2.2 Data Packet Transmission Phase

After a sender finishes transmitting $n^{\mathrm{pri}}_{\mathrm{S}}$ primary data packets to its intended receiver, it immediately switches its role to anticipate the incoming appended packets. As for the potential appenders, each of them checks whether its previous request has been granted by looking up the grant decisions in the CTA packet. A "non-granted" appender i (e.g., node "A_3" in Fig. 5.1) shall remain silent for a duration of $d^{\mathrm{CTA}}_{\mathrm{silent},i}$. In contrast, each granted appender i can start to transmit its granted number of secondary data packets after the specified wait-to-append duration, $d_{\mathrm{wta},i}$, has passed. After transmitting its share of appended packets, it shall remain silent if necessary, so that it does not disrupt the sender's reception of packets from other appenders. For this purpose, each granted appender will locally compute its handshake release duration (relative to the time at which it finishes receiving the CTA) as $d^{\mathrm{CTA}}_{\mathrm{silent},i}$ using (5.2).

Note that an appender does *not* reserve its floor for the entire duration of its secondary transmissions. Instead, its neighboring nodes only need to remain silent for a short duration upon overhearing an RTA to allow the appender to fully receive its CTA. Although the appender will know its secondary transmissions' timings after receiving the CTA, it does not broadcast any control packet again to reserve its floor. Our design rationale is explained as follows. The wait-to-append duration of an appender is typically quite long because it needs to wait for both the primary transmissions as well as some other secondary transmissions to complete.

More importantly, an appender might only be granted to append a few packets as there are multiple appenders in each handshake. Consequently, poor channel utilization may result if each appender were to reserve its floor for an overly long duration, and yet only try to avoid collision for several packets. Due to the fact that an appender does not fully reserve the floor, its secondary transmissions may potentially interfere with the packet receptions of its neighbors. To reduce the likelihood of excessive data collisions, each appender pays close attention to all control packets that it overhears before it begins its packet appending, and will *abort* the scheduled appending if necessary (to be detailed in Section 5.3.5).

One of the salient features of ROPA is its versatile MAC framework; it can operate in three possible data transmission modes, depending on what the initiating sender receives after sending its RTS. In mode 1, a sender receives the CTS, and at least one RTA request; it thus performs both primary and secondary transmissions. In mode 2, it only receives the CTS, and thus it performs primary transmissions only, without having to broadcast a CTA packet; this reduces to the conventional MACA with packet train. In mode 3, it does not receive the CTS, but receives at least one RTA request; since the handshake becomes "receiver"-initiated as the initiating sender becomes a pure receiver, the transmission mode effectively reduces to RIPT. This flexibility results in much better channel utilization since the initiating sender does not waste the time already spent on collecting all the RTA requests and the CTS.

A key mechanism in ROPA for improving channel utilization in underwater acoustic network is that, multiple appenders can be scheduled to transmit with partial overlap in time, such that the appended packets arrive at the initiating sender in a packet train manner without overlapping. This is not possible in terrestrial wireless networks where the propagation delays are usually too short to allow any concurrent transmission. Since ROPA relies on the inter-nodal propagation delay estimates to work, it is essential to protect against any estimation errors; this is achieved via the insertion of a small guard time, T_{guard}, between packet bursts arriving at sender "S" from *different* appenders (see Fig. 5.1). To account for the maximum expected error in the propagation delay estimates, a reasonable value for T_{guard} would be in the range of tens of milliseconds [55].

1 Sort $\tau_{\mathrm{S},i}$ in ascending order for all $i \in \mathcal{N}_{\mathrm{S}} \setminus \{\mathrm{R}\}$.
2 $t^{\mathrm{TX}}_{\mathrm{RTA}}(0) \leftarrow 0; t^{\mathrm{RX}}_{\mathrm{RTS}}(0) \leftarrow 0; \tau_{\mathrm{S}}(0) \leftarrow 0;$
3 **for** $k \leftarrow 1$ **to** $|\mathcal{N}_{\mathrm{S}} \setminus \{\mathrm{R}\}|$
4 $\quad t^{\mathrm{TX}}_{\mathrm{RTA}}(k) \leftarrow \max(t^{\mathrm{RX}}_{\mathrm{RTS}}(k), (t^{\mathrm{TX}}_{\mathrm{RTA}}(k-1) + \tau_{\mathrm{s}}(k-1) - \tau_{\mathrm{s}}(k)$
$\qquad\qquad +T_{\mathrm{RTA}} + T_{\mathrm{guard}}));$
5 $\quad$ **if** $[t^{\mathrm{TX}}_{\mathrm{RTA}}(k) + \tau_{\mathrm{s}}(k),\ t^{\mathrm{TX}}_{\mathrm{RTA}}(k) + \tau_{\mathrm{s}}(k) + T_{\mathrm{RTA}}] \bigcap$
$\qquad [2\tau_{\mathrm{S,R}} + T_{\mathrm{RTS}} - T_{\mathrm{guard}},\ 2\tau_{\mathrm{S,R}} + T_{\mathrm{RTS}} + T_{\mathrm{CTS}} + T_{\mathrm{guard}}] \neq \emptyset$
6 $\qquad t^{\mathrm{TX}}_{\mathrm{RTA}}(k) \leftarrow 2\tau_{\mathrm{S,R}} + T_{\mathrm{RTS}} + T_{\mathrm{CTS}} - \tau_{\mathrm{s}}(k) + T_{\mathrm{guard}};$
7 $\quad d_{\mathrm{dtr}}(k) \leftarrow t^{\mathrm{TX}}_{\mathrm{RTA}}(k) - (\tau_{\mathrm{s}}(k) + T_{\mathrm{RTS}});$

Figure 5.2: Algorithm for scheduling collision-free RTA requests.

5.3.3 Scheduling Algorithms in the ROPA Protocol

5.3.3.1 Algorithm for Scheduling Collision-free RTA Requests

Fig. 5.2 shows how a sender can schedule collision-free RTA requests. In line 1, the sender sorts the inter-nodal propagation delays between itself and all its first-hop neighbors (excluding the receiver) in ascending order. From lines 3 to 7, it calculates the transmit time and the defer-to-request duration for each potential appender. Specifically, in line 4, the transmit time of the k^{th}-order RTA is tentatively set to either immediately after receiving the RTS if it will not collide with a prior potential appender's RTA, or deferred to the earliest time instant when it can do so. Since the RTA must also avoid collision with the receiver's CTS, line 5 checks this condition, and defers the transmit time further, if necessary, in line 6. Finally, in line 7, we calculate the defer-to-request duration for the k^{th}-order potential appender, $d_{\mathrm{dtr}}(k)$.

Recall that t_{ref} is the reference time at which a sender starts sending its CTA. It can be calculated once we have scheduled all the transmit times of the potential appenders:

$$t_{\mathrm{ref}} = \begin{cases} t^{\mathrm{TX}}_{\mathrm{RTA}}(n) + \tau_{\mathrm{S}}(n) + T_{\mathrm{RTA}} + T_{\mathrm{guard}}, & \text{if RTA is last,} \\ 2\tau_{\mathrm{S,R}} + T_{\mathrm{RTS}} + T_{\mathrm{CTS}} + T_{\mathrm{guard}}, & \text{if CTS is last,} \end{cases} \tag{5.3}$$

where $n = |\mathcal{N}_{\mathrm{S}} \setminus \{\mathrm{R}\}|$.

5.3.3.2 Algorithm for Assigning Secondary Data Slots

Let us define two regions within a sender's first-hop neighborhood, which we call regions R_1 and R_2. Here, R_1 is defined as the region in which a node can only communicate directly with the sender but not the receiver. In contrast, R_2 is defined as the region in which a node can communicate directly with both the sender and the receiver.

Using Fig. 5.3, we explain how an initiating sender determines *when* and *how many* data packets a potential appender should append. In line 1, we assume that the sender receives N appending requests, and each request is organized in the form [node ID, $n^{\text{req}}_{\text{relay},i}$, $n^{\text{req}}_{\text{new},i}$, region]. The sender will assign a random priority to these N entries so that an ordered list is created. The stopping condition of the algorithm, as shown in line 3, is either: (i) when all S_{sec} slots are exhausted, or (ii) when slot allocations have been completed for all N entries. Although the ROPA fairness problem remains as future work, we point out that one can achieve fairer allocation by assigning priority based on past request information, as such each appender is given fair amount of time to append [68]. An initiating sender can also prioritize a potential appender with better link quality (i.e., this can be measured upon receiving RTA and CTS requests).

We now discuss how to obtain the wait-to-append duration. To avoid any transmit-receive (TX-RX) collision at the sender, the appended packets must only arrive after it finishes transmitting its primary packets. Furthermore, the appended packets from different appenders must not result in any receive-receive (RX-RX) collision. These are achieved by ensuring that

$$2\tau_{\text{S}}(k-1) + d_{\text{wta}}(k-1) + n^{\text{sec}}(k-1)T_{\text{DATA}} + T_{\text{guard}} \leq 2\tau_{\text{S}}(k) + d_{\text{wta}}(k), \quad (5.4)$$

where $k = \{1, 2, \ldots, N\}$, $d_{\text{wta}}(k)$ is the wait-to-append duration for the k^{th}-order node, and $n^{\text{sec}}(k-1)$ is the number of secondary slots granted to the $(k-1)^{\text{th}}$-order node. More importantly, when the propagation delay between the k^{th}-order node and the sender is sufficiently large, such that $2\tau_{\text{S}}(k) > 2\tau_{\text{S}}(k-1) + d_{\text{wta}}(k-1) + n^{\text{sec}}(k-1)T_{\text{DATA}} + T_{\text{guard}}$, there will be an *idle gap* in the appended packet train received by the sender when the k^{th}-order node's packets arrive (even if $d_{\text{wta}}(k) = 0$), which reduces channel utilization and throughput. For this reason, our algorithm always seeks to avoid this inefficiency by *de-prioritizing* the current

1 All N appending request entries are randomized to create an ordered list, $\mathcal{R}(k)$, where $k \in \{1, \cdots, N\}$.
2 $s_{\text{sec}} \leftarrow S_{\text{sec}}; n^{\text{sec}}(0) \leftarrow n_{\text{S}}^{\text{pri}}; \tau_{\text{S}}(0) \leftarrow 0; d_{\text{wta}}(0) \leftarrow 0;$
$k \leftarrow 1; j \leftarrow 1;$ dp_flag $\leftarrow$ FALSE;
3 **while** $s_{\text{sec}} > 0$ **OR** $k \leq N$
4 **if** $n^{\text{sec}}(j-1) > 0$
5 **if** $2\tau_{\text{S}}(k) \leq 2\tau_{\text{S}}(k-1) + d_{\text{wta}}(k-1) + n^{\text{sec}}(k-1)T_{\text{DATA}} + T_{\text{guard}}$
6 $d_{\text{wta}}(k) \leftarrow 2\tau_{\text{S}}(k-1) + d_{\text{wta}}(k-1) + n^{\text{sec}}(k-1)T_{\text{DATA}} + T_{\text{guard}} - 2\tau_{\text{S}}(k);$
7 **if** $n_{\text{S}}^{\text{pri}} > 0$ **AND** k^{th}-order PA $\neq R$ **AND** from R_2
8 **if** $d_{\text{wta}}(k) < \tau_{\text{S,R}} + n_{\text{S}}^{\text{pri}}T_{\text{DATA}} + T_{\text{guard}} - \tau_{\text{S}}(k) - \tau_{\text{R}}(k)$
9 dp_flag $\leftarrow$ TRUE;
10 **if** $n_{\text{S}}^{\text{pri}} > 0$ **AND** k^{th}-order PA $= R$
11 **if** $d_{\text{wta}}(k) < n_{\text{S}}^{\text{pri}}T_{\text{DATA}} + T_{\text{guard}}$
12 dp_flag $\leftarrow$ TRUE;
13 **else**
dp_flag $\leftarrow$ TRUE;
14 **if** dp_flag = TRUE
15 **if** not cycle through all remaining entries in $\mathcal{R}(k)$
16 de-prioritize current k^{th}-entry to last, goto line 3;
17 **else**
$d_{\text{wta}}(k) \leftarrow 0;$
18 **if** $n_{\text{S}}^{\text{pri}} > 0$ **AND** k^{th}-order PA $\neq R$ **AND** from R_2
19 **if** $\tau_{\text{S}}(k) + \tau_{\text{R}}(k) < \tau_{\text{S,R}} + n_{\text{S}}^{\text{pri}}T_{\text{DATA}} + T_{\text{guard}}$
20 $d_{\text{wta}}(k) \leftarrow \tau_{\text{S,R}} + n_{\text{S}}^{\text{pri}}T_{\text{DATA}} + T_{\text{guard}} - \tau_{\text{S}}(k) - \tau_{\text{R}}(k);$
21 **if** $n_{\text{S}}^{\text{pri}} > 0$ **AND** k^{th}-order PA $= R$
22 $d_{\text{wta}}(k) \leftarrow n_{\text{S}}^{\text{pri}}T_{\text{DATA}} + T_{\text{guard}};$
23 **else**
$d_{\text{wta}}(k) \leftarrow 0;$
24 $n_{\text{relay}}^{\text{grant}}(j) \leftarrow \min[s_{\text{sec}}, n_{\text{relay}}^{\text{req}}(k)]; s_{\text{sec}} \leftarrow s_{\text{sec}} - n_{\text{relay}}^{\text{grant}}(j);$
25 $s_{\text{relay}} \leftarrow 0; s_{\text{rem}} \leftarrow 0;$
26 **for** $i \leftarrow k+1$ **to** N
27 $s_{\text{relay}} \leftarrow s_{\text{relay}} + n_{\text{relay}}^{\text{req}}(i);$
28 $s_{\text{rem}} \leftarrow s_{\text{sec}} - s_{\text{relay}};$
29 **if** $s_{\text{rem}} > 0$
30 $n_{\text{new}}^{\text{grant}}(j) \leftarrow \min[s_{\text{rem}}, n_{\text{new}}^{\text{req}}(k)]; s_{\text{sec}} \leftarrow s_{\text{sec}} - n_{\text{new}}^{\text{grant}}(j);$
31 **else**
$n_{\text{new}}^{\text{grant}}(j) \leftarrow 0;$
32 **if** $n_{\text{relay}}^{\text{grant}}(j) > 0$ **OR** $n_{\text{new}}^{\text{grant}}(j) > 0$
33 $d_{\text{wta}}(j) \leftarrow d_{\text{wta}}(k); n^{\text{sec}}(j) \leftarrow n_{\text{relay}}^{\text{grant}}(j) + n_{\text{new}}^{\text{grant}}(j); j \leftarrow j+1;$
34 $k \leftarrow k+1;$ dp_flag $\leftarrow$ FALSE; remove current entry from $\mathcal{R}(k)$;

Figure 5.3: Algorithm for assigning secondary data slots. "PA" is an abbreviation for potential appender. This algorithm comprises two main subroutines: in lines 4–23, a sender first attempts to find a valid wait-to-append durations; it then allocates its available S_{sec} data slots in lines 24–33.

entry to the last (i.e., "dp_flag" is set to TRUE in line 13), and proceeds to evaluate the next entry. For an entry where $2\tau_{\mathrm{S}}(k) \leq 2\tau_{\mathrm{S}}(k-1) + d_{\mathrm{wta}}(k-1) + n^{\mathrm{sec}}(k-1)T_{\mathrm{DATA}} + T_{\mathrm{guard}}$, as depicted in line 5, there would be no gap between appended packet bursts from different appenders (apart from a small guard time). In this case, $d_{\mathrm{wta}}(k)$ is computed from (5.4) by making its LHS equal to its RHS, as in line 6.

In lines 7–9, if a k^{th}-order potential appender is located in R_2, it must be further ensured that the secondary transmissions will not interfere with the receiver's reception of the primary packets. Specifically, the sender should evaluate whether the previously computed $d_{\mathrm{wta}}(k)$ satisfies the following inequality[2]

$$\tau_{\mathrm{S,R}} + n_{\mathrm{S}}^{\mathrm{pri}} T_{\mathrm{DATA}} + T_{\mathrm{guard}} \leq \tau_{\mathrm{S}}(k) + d_{\mathrm{wta}}(k) + \tau_{\mathrm{R}}(k). \tag{5.5}$$

If it does not, the sender will also de-prioritize the current entry, and consider the next entry. Similarly, in lines 10–12, for the case where the intended receiver wishes to append its packets, its $d_{\mathrm{wta}}(k)$ must be sufficiently large to accommodate its primary packet's receptions. This is achieved by satisfying

$$d_{\mathrm{wta}}(k) > n_{\mathrm{S}}^{\mathrm{pri}} T_{\mathrm{DATA}} + T_{\mathrm{guard}}. \tag{5.6}$$

Lines 14–16 deal with the gap problem by de-prioritizing the current k^{th}-order request, and evaluate the next entry. Note that there would definitely be a gap if all remaining requests still cannot meet the aforementioned conditions. Thus, in line 17, the sender tentatively sets $d_{\mathrm{wta}}(k) = 0$. In lines 18–22, if the k^{th}-order potential appender is either an intended receiver node, or a non-intended receiver node located in R_2, the sender would perform any deferments to $d_{\mathrm{wta}}(k)$, if necessary. Finally, in the absence of primary packets ($n_{\mathrm{S}}^{\mathrm{pri}} = 0$ in mode 3), there will certainly have a gap, thus $d_{\mathrm{wta}}(k)$ is set to zero in line 23.

After obtaining a valid $d_{\mathrm{wta}}(k)$, the sender can allocate data slots from its quota of S_{sec} to this k^{th}-order node using a strategy that prioritizes all relayed packets over new packets, in lines 24–33. This is because the relayed packets have already consumed valuable channel resources, and it would be wasteful if they were

[2]Note that, to evaluate (5.5), the sender "S" must know the estimated inter-nodal propagation delays between its first- and second-hop neighbors (i.e., $\tau_{\mathrm{R}}(k)$), as discussed in Section 5.2. Nevertheless, each node only requires a *subset* of this second-hop neighbor information because the inter-nodal propagation delays of $\tau_{\mathrm{R},j}$, $\forall j \in \mathcal{N}_{\mathrm{R}} \setminus \{\mathcal{N}_{\mathrm{S}} \cap \mathcal{N}_{\mathrm{R}}\}$ are not useful to sender "S".

to be discarded due to potential buffer overflows. Particularly, the sender will first accommodate for the demand of all relayed packets in lines 24–28; then, it only attends to the new packets, if there still have any remaining slots to be allocated, in lines 29–31. After the slot allocations, the sender repeats the aforementioned procedures for the $(k+1)^{\text{th}}$-order node. Finally, let $\mathcal{A}_{\text{G}}$ denote the set of granted appenders; the sender's handshake release duration, $d_{\text{rel,S}}$, is

$$d_{\text{rel,S}} = T_{\text{CTA}} + 2\tau_{\text{S}}(i^*) + d_{\text{wta}}(i^*) + n^{\text{sec}}(i^*)T_{\text{DATA}}, \tag{5.7}$$

where $i^* = \arg\max_{i \in \mathcal{A}_{\text{G}}}[2\tau_{\text{S}}(i) + d_{\text{wta}}(i) + n^{\text{sec}}(i)T_{\text{DATA}}]$.

5.3.4 RTS Attempt Triggering and Backoff Algorithms

In ROPA, we assume that each node maintains two separate buffers for each of its first-hop neighbors; one for relayed traffic, and the other for new (self-originated) traffic. The relayed traffic is prioritized over new traffic because the former has already consumed some network resources previously. Each node uses a hybrid of "*batch-by-size*" and "*batch-by-time*" strategies to determine when to trigger an RTS attempt. More specifically, an RTS attempt will be triggered either when a node has not triggered its RTS attempt for a duration of $T_{\max}$ from the time it last releases from handshake (i.e., batch-by-time), or when it has accumulated at least S_{pri} data packets that are destined for a particular neighboring node (i.e., batch-by-size). Note that an initiating sender can only transmit a maximum of S_{pri} data packets for every handshake, even if it has accumulated more than S_{pri} packets. The composition of S_{pri} may consist of both relayed traffic and new traffic. Even after satisfying either one of the above two triggering conditions, an RTS attempt will only occur if the node is not currently being constrained to remain in a silent state, or engaging in any other handshake. Otherwise, it must defer its RTS attempt until these constraints no longer hold.

The ROPA protocol does not adopt the conventional Binary Exponential Backoff (BEB) algorithm, where a sender doubles its backoff counter in the event of an RTS failure (i.e., the sender does not receive the CTS reply from its intended receiver). When an RTS fails, the ROPA's sender may still be able to receive appended packets. Note that the probability of having no appender is very low, especially when operating at high load. Hence, by the time the sender finishes

receiving all appended packets, the intended receiver may already be free; if the BEB were used, the sender's backoff window could become unnecessarily large, which results in lower throughput. Since the investigation of an optimal backoff algorithm is beyond the scope of this study, we adopt a simple backoff algorithm for ROPA. The ROPA's backoff interval, T_{bk}, is taken from

$$T_{\text{bk}} = \mathit{uniform}(0, B_{\text{win}}) \times \tau_{\max}, \tag{5.8}$$

where B_{win} is a constant backoff window, and $\tau_{\max}$ is the maximum propagation delay. Upon satisfying either one of the two RTS triggering conditions, an idle node will initialize its contention timer according to T_{bk}, and only broadcast its RTS packet upon the timer expiry.

5.3.5 Resolving Potential Problematic Scenarios in ROPA

We have identified three problematic scenarios that may arise in underwater networks. Unless properly addressed, these scenarios would adversely affect the protocol's performance.

Scenario A: Fig. 5.4(a) shows a scenario where an appender A that appends packets to an initiating sender S_1, also falls within the communication range of another initiating sender S_2 as well, but both S_1 and S_2 are hidden from each other. Note that in the figure, packets labeled as "P" are primary data packets, while those labeled as "S" are secondary data packets. As illustrated, if appender A were to simply append its packets, it is likely to cause consecutive data collisions at S_2, when the latter is receiving its secondary data packets. We call this the "*Appending-Induced Data Collision (AIDC)*" problem. This problem would seriously deteriorate ROPA's performance, especially since the appender does not reserve its floor for the secondary data transmissions.

To address the problem, an appender adopts a *listen-before-append* strategy, and aborts its packet appending upon detecting that it is likely to cause the AIDC problem at its neighbors. As shown in Fig. 5.4(b), after A sends its RTA request, it listens closely for any overheard xRTS, xCTS, and xCTA packets[3] during both WFCTA (state in which a potential appender waits for the CTA from its initiating sender) and WTAPP (state in which a granted appender waits

[3]Note that the label "x" implies that the control packet is destined to others. We shall follow this convention for the rest of this chapter.

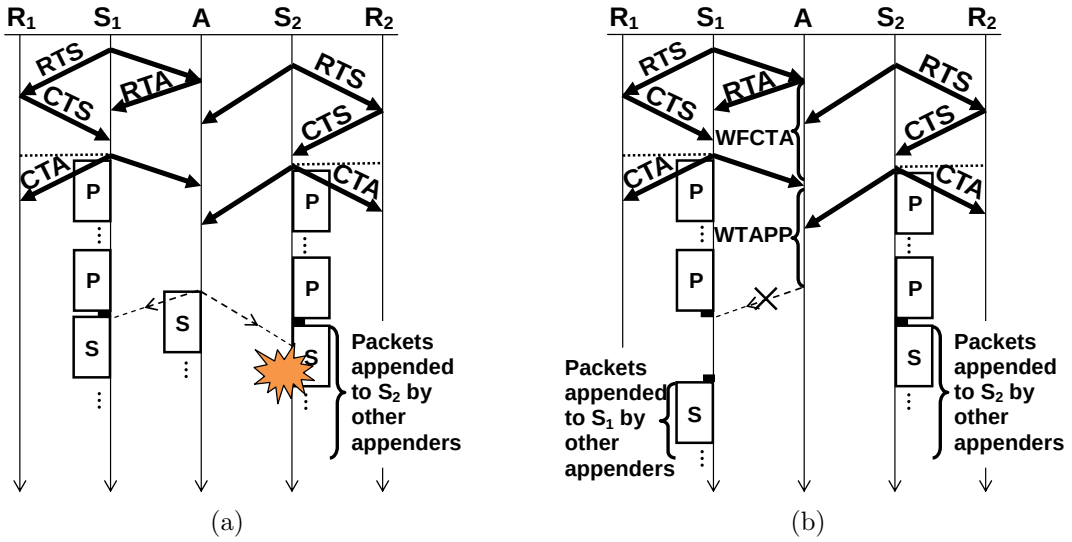


Figure 5.4: (a) Transmission pattern in Scenario A causes appending-induced data collision problem at S_2, (b) proposed solution for Scenario A. Note that packets labeled as "P" are primary data packets, while those labeled as "S" are secondary data packets.

to append its packets) states, as they indicate that the AIDC problem is likely to occur at another neighboring sender or receiver. Although there will be some wasted reserved slots at S_1, it can still receive the remaining appended packets from other granted appenders.

Scenario B: Fig. 5.5(a) shows a problematic scenario in which two neighboring senders, S_1 and S_2, transmit their RTS at around the same time. Since both senders are likely to transmit different number of primary data packets, the primary transmissions from one of the senders can potentially interfere with the secondary packets' receptions at the other sender. Fig. 5.5(b) shows our proposed solution; upon detecting that Scenario B has occurred, both S_1 and S_2 shall revert to the conventional MACA with packet train approach, in which they only transmit their primary data packets. To achieve this, a sender must pay close attention to any overheard xRTS packet during the WFRTA_CTS state, i.e., state in which it waits for the RTA and CTS packets. This indicates that another sender is within its vicinity, and may potentially lead to the data collision problem. Note that the sender can directly send its data packet train, without having to broadcast a CTA packet.

Scenario C: Fig. 5.6(a) shows a scenario where a deadlock may occur when two neighboring initiating senders repeatedly send RTS to each other, albeit the presence of a random backoff mechanism. This problem can be easily resolved

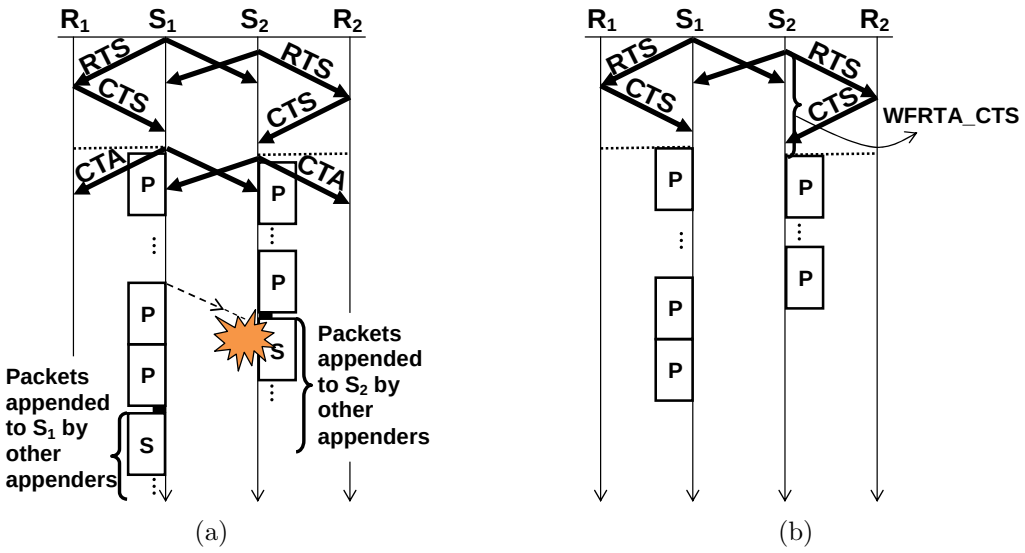


Figure 5.5: (a) Transmission pattern in Scenario B may result in consecutive data collisions at S_2, (b) proposed solution for Scenario B.

as follows. Recall that in the RTS packet, a sender will announce the number of relayed and new packets that it intends to transmit. Hence, a sender can figure out its priority level against the other node according to a set of priority rules. A node first compares the total number of packets it intends to transmit with what the other node has indicated in its RTS. In the case of identical priority, it can further compare the total number of relayed packets. If both senders still have identical priority, then this can be resolved based on whose unique ID is larger. In Fig. 5.6(b), we assume that S_2 wins the priority test, and hence S_1 replies with a CTS. Although S_1 has become a receiver node, it can still request for packet appending (via using its CTS reply) to transmit some data packets to S_2.

5.3.6 Adaptive Primary and Secondary Packet Train Sizes

In practice, an adaptive mechanism to assemble primary and secondary packet trains is desired, because it is unlikely that a single set of static $[T_{\max}, S_{\text{pri}}, S_{\text{sec}}]$ parameters would perform well across all network loads.

5.3.6.1 Adjusting Primary Train Sizes

We propose a mechanism to dynamically adjust S_{pri} based on the run-time network loading measurements, while keeping $T_{\max}$ fixed. The network loading can be estimated using the queue occupancy measurement technique [59]. We shall keep S_{pri} as small as possible when network load is low, and allow it to increase as the

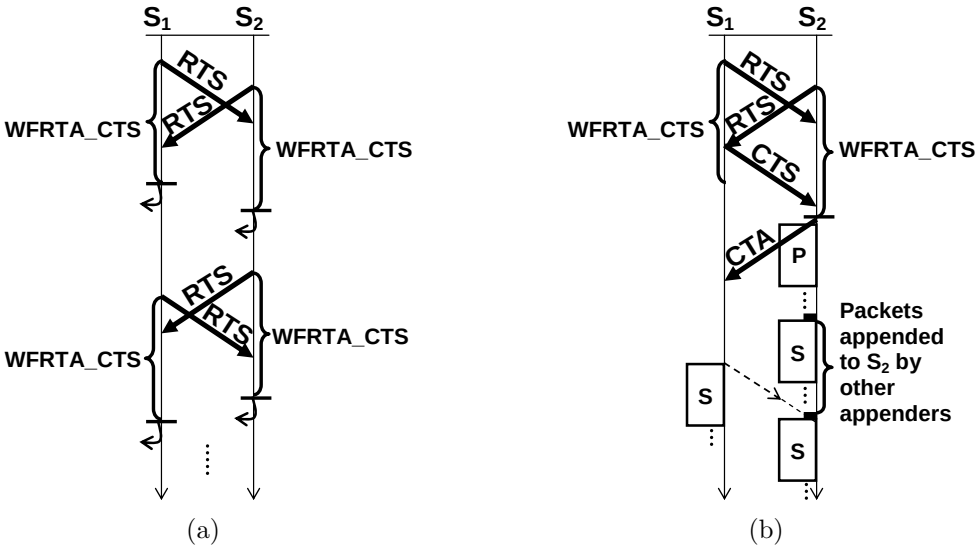


Figure 5.6: (a) Transmission pattern in Scenario C may result in a deadlock, (b) proposed solution for Scenario C.

load grows. So, the "batch-by-size" mechanism becomes the main RTS triggering mechanism across all network loads. Similar to [56] and [61], $T_{\max}$ is fixed to a pre-defined constant, and its value depends on the higher protocol layers. The reason is that, S_{pri} could occasionally be overestimated because the network load changes dynamically; this causes the delay performance to deteriorate due to the longer time taken to form the larger packet trains. Hence, $T_{\max}$ is used as a safeguard mechanism to ensure that the packet delays do not grow unrestrained.

Recall that each node uses two separate buffers to store relayed and new packets for each of its N first-hop neighbors. In our mechanism, we partition time into intervals of equal duration. Thus, each node keeps track of the average number of data packets for all its buffers (i.e., $Q_{x,i}(k)$, where $x \in \{\text{relay}, \text{new}\}, i \in \{1, \ldots, N\}$) in the k^{th}-interval, where $k \in \mathbb{Z}^+$. At the end of the k^{th}-interval, a node computes its normalized queue occupancy for each neighboring node i:

$$Q_i(k) = \frac{Q_{\text{relay},i}(k) + Q_{\text{new},i}(k)}{Q_{\text{relay}}^{\max} + Q_{\text{new}}^{\max}},\ i \in \{1, \cdots, N\}, \tag{5.9}$$

where $Q_x^{\max},\ x \in \{\text{relay}, \text{new}\}$ denotes the maximum buffer size for the respective type of packets. Then, the average traffic intensity indicator for the current k^{th}-interval is:

$$I_{\text{avg}}(k) = \frac{\sum_{i=1}^{N} Q_i(k)}{N}. \tag{5.10}$$

In order to smoothen out the random fluctuations, an exponential moving average is obtained as follows:

$$\overline{I_{\text{ma}}}(k) = \alpha \cdot \overline{I_{\text{ma}}}(k-1) + (1-\alpha) \cdot I_{\text{avg}}(k), \tag{5.11}$$

where α is a smoothing factor set to 0.9. We use this constant because it has been shown that it is a good compromise between accuracy and promptness [62]. Note that $\overline{I_{\text{ma}}}(0) = 0$. Finally, the estimated S_{pri} for the next interval is computed as

$$S_{\text{pri}}(k+1) = \left\lfloor S_{\text{pri}}^{\min} + \overline{I_{\text{ma}}}(k) \cdot [S_{\text{pri}}^{\max} - S_{\text{pri}}^{\min}] \right\rfloor, \tag{5.12}$$

where $S_{\text{pri}}^{\min}$ and $S_{\text{pri}}^{\max}$ are the respective minimum and maximum S_{pri} values. Here, $S_{\text{pri}}^{\max}$ is a fixed value that is subjected to buffer constraints, while $S_{\text{pri}}^{\min}$ is set to the smallest unit, i.e., $S_{\text{pri}}^{\min} = 1$. Whenever a node releases from a handshake, it uses the most recently estimated S_{pri} as a threshold to initiate its next RTS attempt. All nodes start with $S_{\text{pri}}(1) = 1$ and they adapt their own S_{pri} independently.

5.3.6.2 Adjusting Secondary Train Sizes

We now explain how an initiating sender dynamically tunes its S_{sec} parameter. Specifically, for a given handshake loop n, the sender could sum up the total number of secondary packets that each of its neighbors wishes to append, $n_{\text{req}}(n)$. Then, at time instant t_{ref}, the sender utilizes this information to predict its *next* handshake loop's secondary train sizes, $S_{\text{sec}}(n+1)$, as follows:

$$S_{\text{sec}}(n+1) = \begin{cases} \min[2S_{\text{sec}}(n), S_{\text{sec}}^{\max}], \text{if } n_{\text{req}}(n) \geq S_{\text{sec}}(n), \\ \max[S_{\text{sec}}(n)-1, S_{\text{sec}}^{\min}], \text{if } n_{\text{req}}(n) < S_{\text{sec}}(n), \end{cases} \tag{5.13}$$

where $S_{\text{sec}}^{\min}$ and $S_{\text{sec}}^{\max}$ are the respective minimum and maximum S_{sec} values. Here, $S_{\text{sec}}^{\min}$ is set to 1, and $S_{\text{sec}}^{\max}$ is a fixed value that is subjected to buffer constraints. All nodes start with $S_{\text{sec}}(1) = 1$ and adapt their S_{sec} independently. When $n_{\text{req}}(n) < S_{\text{sec}}(n)$, $S_{\text{sec}}(n+1)$ is subjected to a more cautious linearly decreasing strategy, since the observed $n_{\text{req}}(n)$ could be less than the actual intended data slot requests because the sender may not have successfully received all the RTA and CTS packets. In contrast, when $n_{\text{req}}(n) \geq S_{\text{sec}}(n)$, such uncertainty is definitely

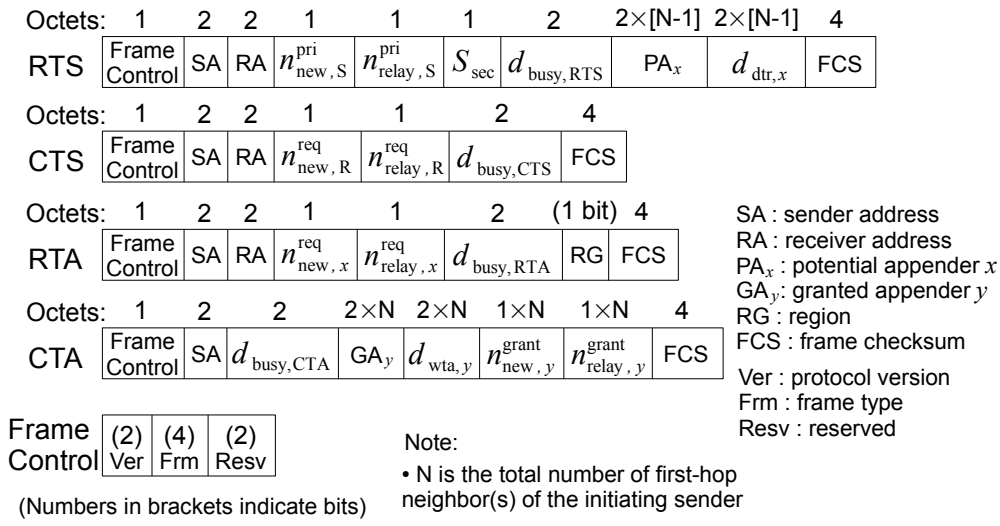


Figure 5.7: Our proposed control packet formats for the ROPA protocol.

not present; hence, a multiplicative increase strategy is adopted. Finally, our proposed control packet formats for ROPA are summarized in Fig. 5.7.

5.4 Performance of ROPA in Multi-hop Networks

5.4.1 Simulation Model

We have developed a custom C++ discrete event-driven network simulator. Our multi-hop topology, shown in Fig. 5.8, has 36 static nodes with a grid spacing of 2000 m. The maximum communication range is 1.75 times the grid spacing, or 3500 m; the interference range is assumed to be the same as the communication range (the effects of large interference range are studied in Section 5.6). Hence, each node has exactly 8 first-hop neighbors and 16 second-hop neighbors. A *wrap-around* strategy is used to distribute network load evenly and eliminate boundary effects. Each node generates its data packets according to the Poisson distribution, and randomly picks one of the 16 second-hop neighbors as a *destination*. Every node has a half-duplex omni-directional transceiver. The channel is also assumed to be error-free unless stated otherwise, so that packet losses are solely caused by packet collisions. The acoustic propagation speed is 1500 m/s, and the link transmission rate is 4800 bps. Unless stated otherwise, the data packet length is 1200 bits, and the lengths of the RTS, CTS, RTA, and CTA packets are set to 336, 104, 105, and 456 bits, respectively. For the guard time and backoff window, we set $T_{\text{guard}} = 10$ ms and $B_{\text{win}} = 15$. For the packet train based protocols, our adaptive mechanism for tuning the packet train sizes, as presented in Section 5.3.6,

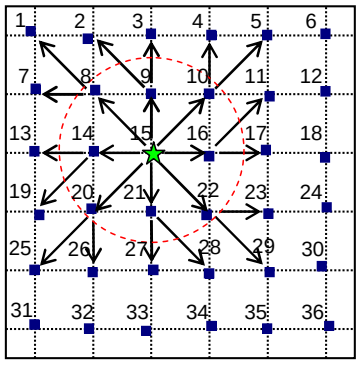


Figure 5.8: The multi-hop network topology used in our simulations. Every other node assumes the same static routing pattern as the node marked with a star. We only consider two-hop routes so that it is easier to interpret the results. Also, each node is allowed to randomly deviate from the grid intersection point by a maximum of 10% of the grid spacing, so as to introduce randomness.

is also employed; we set $T_{\max} = 100$ s, $S_{\text{pri}}^{\min} = S_{\text{sec}}^{\min} = 1$, $S_{\text{pri}}^{\max} = S_{\text{sec}}^{\max} = 200$, $S_{\text{pri}}(1) = S_{\text{sec}}(1) = 1$, $\alpha = 0.9$, and the updating interval is set to 50 s. A node maintains two buffers (for relayed and new packets) for each of its first-hop neighbors, where each buffer can hold 100 packets. Lastly, we do not put any upper limit on the number of retries when RTS attempts fail. To avoid transient effect, the results are collected from 2×10^4 s to 1×10^5 s. Also, all simulation results are averaged over 5 different topologies.

In order to illustrate ROPA's superiority, we compare against MACA-U [66] and MACA-UPT variants, which only perform *primary* data transmissions. As for the performance metrics, we are interested in: (i) normalized throughput per node (end-to-end), and (ii) end-to-end data packet delay, in which they are defined earlier in Section 4.4.2.

5.4.2 Simulation Results

5.4.2.1 Comparison of ROPA against other schemes

Fig. 5.9(a) shows the normalized throughput per node (hereafter referred to as "throughput") for several MACs when we vary the normalized offered load per node; Fig. 5.9(b) shows their average end-to-end data packet delay (hereafter referred to as "delay"). As shown, ROPA and MACA-UPT variants significantly outperform MACA-U in both throughput and delay due to the use of packet train mechanism. As expected, MACA-U's throughput is very low (around 0.001), while ROPA greatly surpasses it (by over 33 times). Clearly, it is costly to transmit only

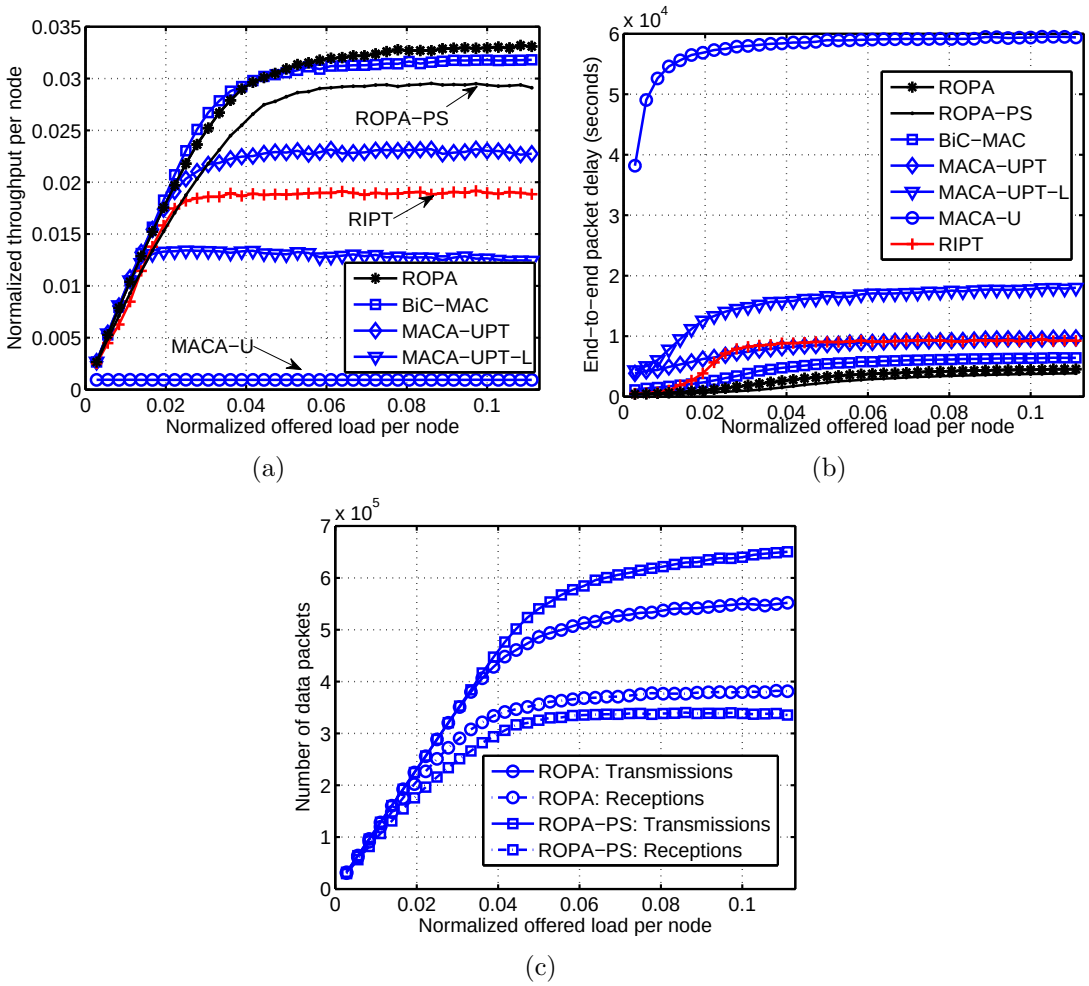


Figure 5.9: Comparisons for various schemes: (a) normalized throughput per node, (b) end-to-end data packet delay, (c) number of data packets transmitted/received.

a single packet per handshake in a long propagation delay environment because the proportion of time spent on the multi-way control packet exchange will become very significant. For this reason, MACA-U has a very large delay as well.

We now compare ROPA with MACA-UPT variants, which only allow primary data transmissions. Recall that ROPA needs the first-hop neighbors of an initiating sender to remain silent upon overhearing an xRTS packet until the end of the secondary data receptions at the sender (except for the respective time window in which the neighbor has been granted permission to append its packets). This requirement may appear inefficient, but it is necessary for ROPA to operate correctly. In contrast, MACA-UPT only requires these first-hop neighbors to remain silent for a relatively shorter duration upon overhearing an xRTS, for accommodating the subsequent CTS. Thus, MACA-UPT benefits from better spatial reuse, since these neighbors can participate in a new handshake sooner.

By comparing the throughput of MACA-UPT with that of MACA-UPT-L, which intentionally extends the silent durations till the end of the packet train just to investigate the effect, we see from Fig. 5.9(a) that the maximum throughput can drop by nearly 45% for the latter. Additionally, in the saturation throughput region, ROPA outperforms MACA-UPT and MACA-UPT-L by around 46% and 166%, respectively. ROPA also has lower delay than them across all offered load ranges, in which the saturation delay is reduced by around 43% and 75%, respectively. This shows that, although ROPA may be inefficient in terms of the longer silent durations, its performance gains arising from the opportunistic packet appending are able to more than compensate for these inefficiencies. Although it still utilizes a sender-triggered RTS/CTS based handshake, multiple backlogged neighbors (including the intended receiver) can use that opportunity to transmit their data packets to the initiating sender. Without such a mechanism, each of those neighbors must initiate their own handshake, which would certainly incur much higher overheads. Also, the packet appending's setup cost is minimal because the time spent on polling/request/grant fully overlaps with the RTS/CTS/Primary DATA handshake durations.

Figure 5.9(a)–5.9(b) show that, at high loads, ROPA has a saturation throughput and delay of around 0.0334 and 4500 seconds, respectively. While the end-to-end delay is high in our default setting, we aim to demonstrate ROPA's *potential* gain compared to existing MACs. We note that the large delay is due to using: (i) low bit-rate of 4800 bps, (ii) dense topology with a grid spacing of 2000 m, (iii) no upper limit on the number of RTS retries, (iv) higher queueing delays from the per-neighbor queues compared to per-node queues, and (v) high traffic loads as every node needs to relay packets for others. Nonetheless, when we modify the bit-rate to 15000 bps and smaller grid spacing of 100 m, we found that the saturation throughput and delay of ROPA improve to around 0.0351 and 847 seconds, respectively.

We also simulate a variant of ROPA that does not handle the problematic scenarios presented in Section 5.3.5, for which we call "ROPA-PS". Compared to ROPA in Fig. 5.9(a)–5.9(b), ROPA-PS has inferior throughput, and its saturation throughput reduces by around 12%. However, ROPA-PS's delay is slightly better, where it has a saturation delay gain of around 15%. To gain more insight, we examine the number of data packets transmitted (n_{tx}) and received (n_{rx}) for

both protocols, as shown in Fig. 5.9(c). We also define a success ratio metric as $n_{\text{rx}}/n_{\text{tx}}$. In the saturation region, the success ratio of ROPA is around 72%, while ROPA-PS only achieves a ratio of 49%. Since ROPA-PS does not abort its packet appending even though it is likely to cause data collisions, it can thus transmit more packets in a given duration, which yields a lower packet delay. However, many of ROPA-PS's packet transmissions are wasted because they encounter more collisions. Although our mechanisms presented in Section 5.3.5 do not guarantee collision-free transmissions, the above results show that they are effective in alleviating the excessive packet collisions.

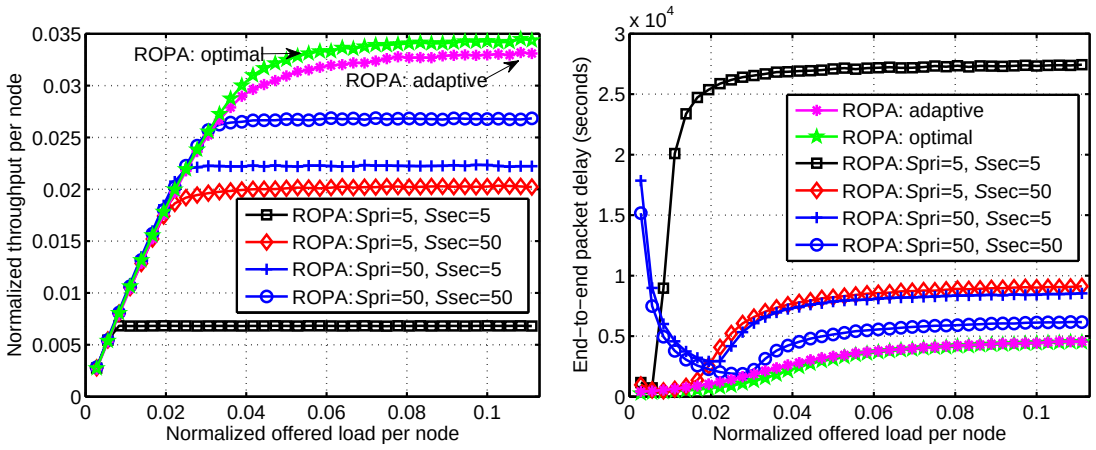
Next, we study the RIPT protocol [27], which relies on a receiver-initiated, multiple-node transmission approach, and is a special case of ROPA that only has the secondary transmissions. Fig. 5.9(a)–5.9(b) show that ROPA offers remarkable improvements over RIPT; specifically, ROPA achieves a saturation throughput gain of around 76%, as well as a saturation delay reduction of 51% over RIPT.

Finally, we compare ROPA against the BiC-MAC protocol [69], which is a sender-initiated handshaking-based MAC protocol that exploits the long propagation delay by using a unique approach of bidirectional-concurrent packet bursting. In each handshake, BiC-MAC only allows a sender-receiver node pair to simultaneously transmit their data packets to each other. In contrast, ROPA allows an initiating sender to schedule multiple appenders to transmit with partial overlap in time, such that those data packets arrive at the sender in a packet train manner. Fig. 5.9(a)–5.9(b) show that ROPA outperforms BiC-MAC, in both saturation throughput and delay.

5.4.2.2 Performance of adaptive train size mechanism

Using Fig. 5.10(a)–5.10(b), we explain why the adaptive train size mechanism is necessary, by showing that the non-adaptive ROPA variants do not work well across all offered load regions. For these non-adaptive variants, their packet train size [S_{pri},S_{sec}] are statically fixed to {[5,5],[5,50],[50,5],[50,50]}, while $T_{\max}$ is assumed to be infinity so that all RTS attempts are triggered solely by the "batch-by-size" mechanism.

As shown, both variants with $S_{\text{pri}} = 5$ (i.e., {[5,5],[5,50]}) perform well in terms of throughput and delay at low load, but their performance deteriorates quickly as the offered load increases. Recall that in the "batch-by-size" mechanism,



(a) (b)

Figure 5.10: Performance comparisons of ROPA with adaptive train size mechanism against several other non-adaptive ROPA variants in terms of: (a) normalized throughput per node, (b) end-to-end data packet delay.

S_{pri} determines when a node triggers its RTS attempt, for which a primary train of S_{pri} packets will be sent. Thus, at high load, a smaller S_{pri} performs poorly due to a larger communication overhead as the sender only transmits a short primary packet train for each handshake (larger proportion of time spent on control packet exchanges). In contrast, both ROPA variants with $S_{\text{pri}} = 50$ (i.e., {[50,5],[50,50]}) perform better at high load, but their packet delays are very high at low load. When packet arrival rate is low, a larger S_{pri} makes a node harder to satisfy the "batch-by-size" requirement; hence, a large delay is resulted due to the longer waiting time for accumulating sufficient packets. In short, regardless of $T_{\max}$ settings, S_{pri} should be kept as small as possible at low load, but allowed to increase accordingly as the load grows.

When total train size is kept constant (i.e., {[5,50],[50,5]}), a larger S_{pri} offers better performance at high load. There are two reasons for this. First, when a receiver's neighbors overhear its CTS, they have long silent durations till the end of its primary packets' receptions. In contrast, an appender's neighbors only remain silent for a short duration upon overhearing its RTA (up to the CTA's reception at the appender), and they may engage in other handshakes after that. Thus, the secondary transmissions from an appender might cause interference to its neighbors' reception of packets subsequently, which reduces throughput. Although an appender could alleviate data collisions by aborting its packet appending, its previously granted slots will be wasted. Second, while a sender will transmit exactly S_{pri} number of packets after its successful RTS

Table 5.2: Saturation Throughput Per Node and End-to-End Packet Delay Comparisons for Different Inter-nodal Distances

Grid	ROPA		MACA-UPT		Throughput	Delay
Sizes	Throughput	Delay	Throughput	Delay	Gain	Reduction
1000 m	0.0350	3612 s	0.0290	7613 s	20.69%	52.55%
3000 m	0.0310	5357 s	0.0185	11782 s	67.57%	54.53%
5000 m	0.0274	6697 s	0.0140	15136 s	95.71%	55.75%
7000 m	0.0245	8010 s	0.0112	18240 s	118.75%	56.09%
9000 m	0.0223	9136 s	0.0091	21865 s	145.05%	58.22%

attempt, the actual number of secondary packets received may not always be exactly S_{sec}. Occasionally, the sender's first-hop neighbors may only intend to transmit a total of less than S_{sec} packets, and thus ROPA only needs to cater for those requests, even if it has a quota of S_{sec}. We also compare against an optimal ROPA scheme, where [$T_{\max}$,S_{pri},S_{sec}] are set to [100 s,130,90] obtained from simulations. As can be seen, the adaptive ROPA's performance is very close to that of the optimal scheme. A key advantage of the adaptive approach is that it can tune its parameters based on real-time loading, as opposed to using simulations to determine the optimal settings.

5.4.2.3 Effects of varying inter-nodal distances

Table 5.2 shows the effects of varying inter-nodal distances (i.e., grid sizes) on ROPA's saturation throughput and delay, when the normalized offered load per node is fixed at 0.1111. Generally, underwater handshaking-based MAC protocols are sensitive to the inter-nodal propagation delay, as a multi-way handshake is needed prior to the data transmissions. As inter-nodal distances grow, the communication overhead is amplified, which leads to performance degradation. However, as shown, when the inter-nodal distances become larger, both ROPA's throughput gain and delay improvement over MACA-UPT become more significant. This is desirable since underwater acoustic networks often cover a wide region, where the inter-nodal distances are typically in the order of several kilometers [1, 7].

5.4.2.4 Effects of varying packet error rate

We explore the effects of imperfect channel, where the control and data packets are subjected to a packet error rate (PER). Our goal is to study the protocol's

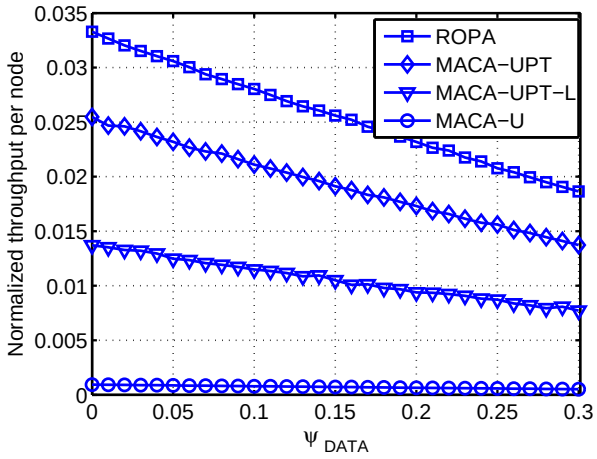


Figure 5.11: Effects of varying packet error rate on ROPA's normalized saturation throughput per node. The normalized offered load per node is fixed at 0.1111.

robustness, since underwater channel is unreliable [1]. In our simulations, each data packet of length L_{DATA} is subjected to a pre-defined PER, say ψ_{DATA}; while each control packet of length L_x, where $x \in \{\text{RTS}, \text{CTS}, \text{RTA}, \text{CTA}\}$, will be subjected to a PER of $\psi_x = (L_x/L_{\text{DATA}}) \cdot \psi_{\text{DATA}}$. Fig. 5.11 shows the effects of varying PER on ROPA's saturation throughput, when the normalized offered load per node is fixed at 0.1111. As shown, the throughputs of all protocols degrade gracefully without any sudden drops. The ROPA protocol also outperforms the rest of the protocols across all simulated PER ranges, offering a throughput of around 0.0186 when $\psi_{\text{DATA}} = 0.3$. ROPA is robust against control packet losses (e.g., due to channel bit errors). When an appender receives a corrupted RTS or CTA packet, the only effect on ROPA is that, this particular node loses its chance to append in the current round of handshake, which reduces the throughput. This, however, does not affect the protocol operation because other neighbors can still append their packets. When a sender does not receive an RTA or CTS correctly from a neighbor, it cannot allocate its secondary data slots to that specific neighbor only, but can still do so for other neighbors. Note that the sender also cannot transmit primary data packets to its intended receiver if the CTS is not successfully received.

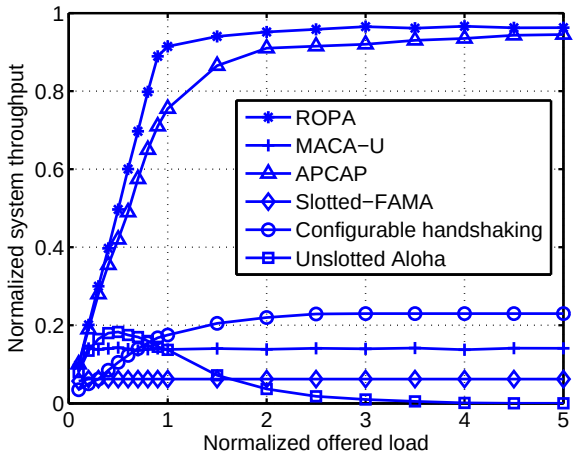


Figure 5.12: Normalized system throughput comparisons of ROPA and MACA-U protocols against several other selected underwater MAC protocols in [20].

5.5 Performance of ROPA in Single-hop Networks

5.5.1 Simulation Model

We now compare ROPA against several other schemes in single-hop networks. Guo *et al.* [20] have compared their proposed APCAP protocol with several selected MAC protocols such as the Slotted-FAMA [19], configurable handshaking protocol (DACAP) [21], and unslotted-Aloha. We follow the same setup, where 20 static nodes are randomly deployed in a 4500×4500 m^2 area. The link transmission rate is 2400 bps, and data packet length is 8000 bits. Each data packet generated is allowed to be destined to any one of the neighbors with equal probability. We also employ the adaptive packet train size mechanism; we set $B_{\text{win}} = 15$, $T_{\max} = 100$ s, $S_{\text{pri}}^{\min} = S_{\text{sec}}^{\min} = 1$, $S_{\text{pri}}^{\max} = S_{\text{sec}}^{\max} = 100$, $S_{\text{pri}}(1) = S_{\text{sec}}(1) = 1$, $\alpha = 0.9$, and the updating interval is 50 s. For packet train based protocols, each node maintains a single buffer of 100 packets for each of its neighbors; while MACA-U only maintains a single buffer with a size of 1900 packets. All results are averaged over 5 different topologies.

5.5.2 Simulation Results

Fig. 5.12 shows that ROPA achieves the highest system throughput, and it performs better than APCAP across all offered load ranges. This shows that the opportunistic packet appending approach is still useful in single-hop networks. As for the rest of the MAC protocols, we see that the configurable handshaking

protocol outperforms Slotted-FAMA. Although Slotted-FAMA employs a time-slotting mechanism to avoid data collisions, its long slot size requirement degrades its throughput. In contrast, the configurable handshaking protocol uses a more efficient handshake, where a sender is allowed to use different handshake lengths for different receivers so that the average handshake duration can be minimized. As expected, the unslotted-Aloha cannot maintain its throughput as the load increases because it does not employ any collision avoidance mechanism. Interestingly, MACA-U performs better than Slotted-FAMA across all offered loads, as well as the configurable handshaking protocol in the low load region, even though MACA-U only transmits a single data packet for each successful handshake.

5.6 Discussion

5.6.1 Enhancing ROPA with Packet Acknowledgement Scheme

To allow retransmission of any unsuccessfully received packets, we enhance ROPA with packet acknowledgement scheme by proposing a novel "*Reverse-Forward Block ACKnowledgement (RF-BACK)*" mechanism. Unlike the conventional Stop-and-Wait Automatic Repeat reQuest (ARQ) variants introduced in [29], which only send a single acknowledgement (ACK) at the end of primary packet receptions, RF-BACK can cater for both primary and secondary data block receptions (i.e., forward and reverse path transmissions), as shown in Figures 5.13(a)–5.13(b).

Each transmitted packet usually carries error control bits like cyclic redundancy check (CRC), which can be used at a receiving node for packet error detection [29]. Upon finish receiving appended data packets from each granted appender, an initiating sender "S" can send a reverse ACK to announce the receiving status of its currently received secondary packets. Using inter-nodal delay information, each granted appender can estimate the ACK packet's arrival time, and thus its silent duration can be extended if needed. Note that the sender should not only send a *single* aggregate ACK, at the end of the entire secondary data receptions, because all appender nodes would have unnecessarily long handshake release time. For primary data receptions, an intended receiver "R" can piggyback forward ACKs in each of its appended packet; redundant ACKs are inserted to improve robustness. Figure 5.13(b) shows that if the receiver

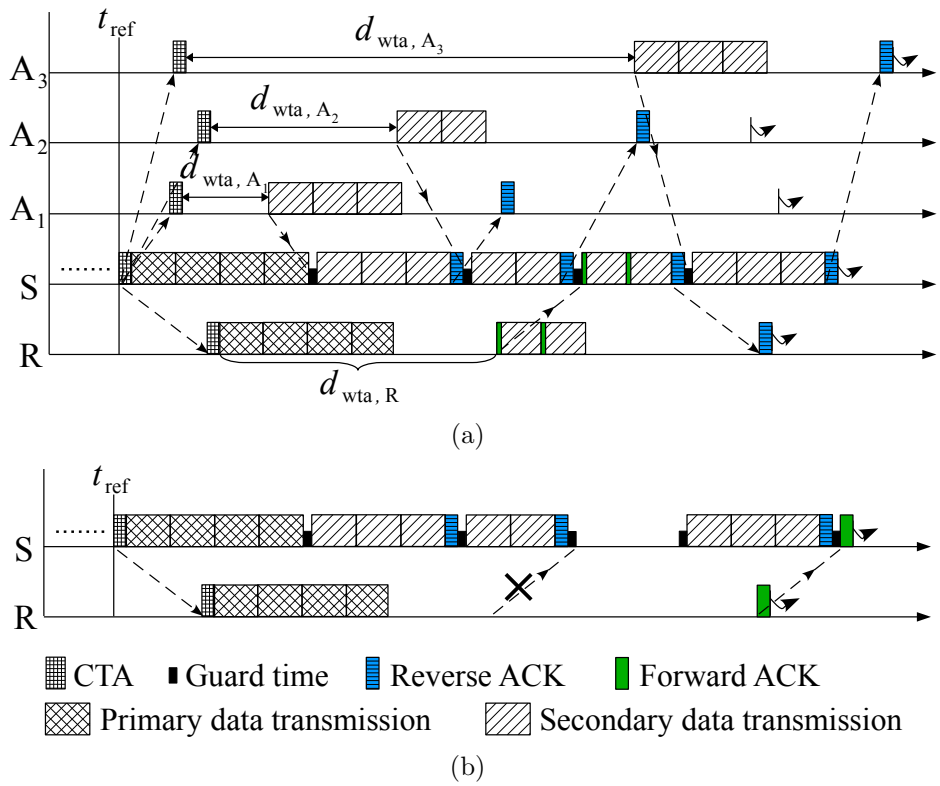


Figure 5.13: ROPA with ACK enhancement: (a) whenever a sender "S" finishes receiving a block of secondary packets from a granted appender, it sends an ACK to report its packet receiving status before starting to receive appended packets from another appender. To notify the primary data reception status, an intended receiver "R" can piggyback its ACK in its secondary packet transmissions; (b) if the receiver "R" does not become an appender, it can schedule a stand-alone ACK transmission at the end of the current handshake.

"R" does not send its appended packets (e.g., when receiver "R" aborts packet appending, or sender "S" does not grant its appending request, etc.), it can still send a forward ACK just before the expiry of current handshake. Note that when ROPA operates in mode 2 (only primary packets), the receiver "R" can immediately send a single ACK at the end of its primary data receptions; for mode 3 operation (only secondary packets), the sender "S" responses with multiple ACKs, similar to Figures 5.13(a).

We point out that when channel is very bad, a transmitted packet can be totally undetected at a receiving node. Thus, we cannot rely on the packet's CRC for error detection. In ROPA, however, sender "S" and receiver "R" know how many packets to be exchanged in each handshake. Hence, even when a packet cannot be successfully detected, a node is still able to conclude that its received packet is lost and that packet will be reported as error in the ACK. When a sending node discovers any unsuccessful packet receptions at its intended receiving node,

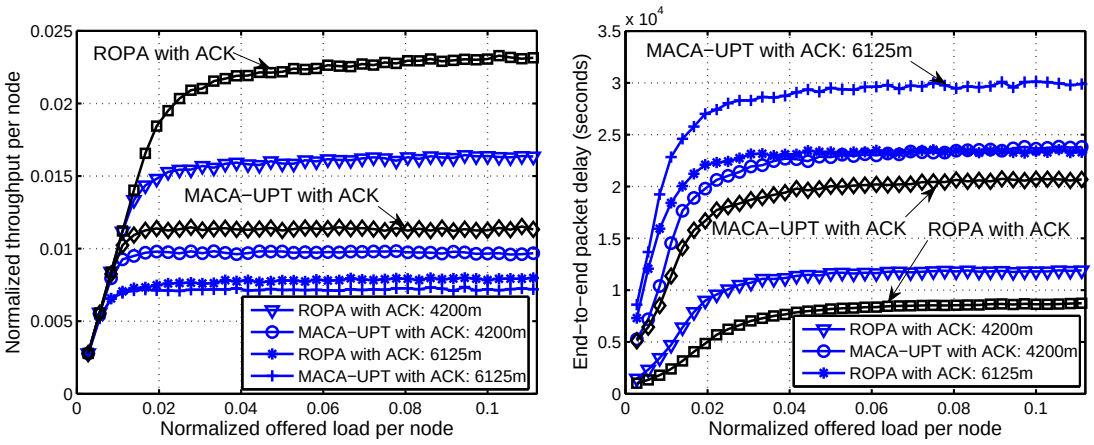
all erroneous packets can be retransmitted along with any accumulated relayed and self-originated traffics, in its next handshake loop. To reduce packet delay, we prioritize those packets that are waiting for retransmission over relayed and self-originated traffics. Note that a node still uses "batch-by-size" and "batch-by-time" mechanisms (Section 5.3.4) to determine its RTS attempt triggering.

We study the effects of using ACKs in ROPA and compare its performance to MACA-UPT with ACK, in throughput per node and end-to-end delay (labeled as "ROPA with ACK" and "MACA-UPT with ACK" in Figures 5.14(a)–5.14(b)). Unless otherwise stated, we use the same settings of 36-node multi-hop network in Section 5.4.1. Each ACK packet length is set to 272 bits. We do not put any limit on the number of retransmissions when data packet is not received correctly. For now, interference range is assumed to be the same as the communication range of 3500 m. With reference to ROPA and MACA-UPT without ACKs in Figures 5.9(a)–5.9(b), the addition of ACK to MACA-UPT causes its saturation throughput and delay to deteriorate by around 50.06% and 112.88%, respectively. Note that when using ACK in MACA-UPT, the first-hop neighbors of an initiating sender must now extend their silent durations to cater for ACK in return, and thus cannot benefit from spatial reuse (similar to the earlier "MACA-UPT-L"). Although utilizing RF-BACK in ROPA degrades its saturation throughput and delay by around 30.46% and 89.37%, respectively, it still greatly outperforms MACA-UPT with ACK in both throughput and delay. Our RF-BACK scheme, which relies on multiple reverse ACKs and redundant forward ACKs, has better retransmission efficiency in the event of ACK loss. This is unlike the MACA-UPT with ACK, in which *all* previously sent data packets (even if received correctly) need to be retransmitted when the single ACK is lost, as the sender is uncertain about the packet receiving status of each data packet at the receiving node.

5.6.2 Effects of Large Interference Range

We evaluate the effects when interference range is larger than communication range[4]; both ROPA and MACA-UPT with ACKs are considered. In the simulation, the interference range is set to 1.20 [67] or 1.75 times the communication range of 3500 m (i.e., 4200 m or 6125 m). Note that in our multi-hop grid

[4]Similar to previous work [34] that evaluates MAC performance in a grid topology, we also focus on a fixed interference range model.



(a) (b)

Figure 5.14: Effects of using ACKs in ROPA and MACA-UPT, and the impacts of large interference range on these MAC protocols with ACKs: (a) normalized throughput per node, (b) end-to-end data packet delay.

topology, the latter range is large enough to cause packet interference to all 16 second-hop neighbors. When interference range is set to 4200 m or 6125 m, the saturation throughput for ROPA with RF-BACK drops by around 29.3% or 65.5%; its corresponding saturation delay also degrades by 36.2% or 166.6%. As interference range grows, it greatly limits ROPA's performance, since increasingly more number of second-hop neighbors around a sending node (i.e., initiating sender or appender) cannot receive packets correctly. Nonetheless, we found that ROPA still offers better throughput and delay performance than MACA-UPT with ACK, under both simulated interference ranges.

5.6.3 Using ROPA Handshake Mechanism to Estimate Inter-nodal Delays

During network initialization, each node first estimates inter-nodal propagation delay from each of its first-hop neighbors using RTT measurements; it then gathers subset of the inter-nodal delay between its first- and second-hop neighbors. Although ROPA uses guard times to cater for uncertainty in sway movements and delay estimation errors, we can improve the protocol robustness by utilizing the inherent handshake framework to continually estimate and correct inter-nodal delay information. In every handshake loop, an initiating sender has two opportunities to measure RTT for each neighboring node, i.e., during: (i) "RTS/RTA", "RTS/CTS", and (ii) "CTA/Secondary DATA". Each neighbor also can measure RTT between itself and the sender during "RTA/CTA", "CTS/CTA", and

"Secondary DATA/Reverse ACK" handshakes. By examining deviations from expected packet arrival times, each node can regularly update its first-hop neighbors' inter-nodal delays. When a node notices a large deviation, it can piggyback updated delay information and broadcast using control packets; this allows neighboring nodes to correct their second-hop neighbor information. While this introduces extra control packet overhead, the accurate inter-nodal delay information allows for better ROPA operation.

5.6.4 Scalability of ROPA

We now discuss the scalability issue. As the number of first-hop neighbor N increase (hence the available potential appenders), the control packet overheads become more significant because both RTS and CTA packets' sizes directly depend on N, as shown in Fig. 5.7. To overcome this, an initiating sender could limit the number of potential appenders in each handshake. More specifically, the sender can divide its N first-hop neighbors into M disjoint groups $\mathcal{G}_i$, $i \in \{1 \dots M\}$, such that $\mathcal{G}_1 \cap \mathcal{G}_2 \dots \cap \mathcal{G}_M = \emptyset$ and $\sum_i |\mathcal{G}_i| = N$. Then, for a given handshake loop k, the sender only polls its potential appenders from nodes in group $\mathcal{G}_1$; subsequently in its next handshake loop $k+1$, it will poll from group $\mathcal{G}_2$, and etc. Via this round-robin group polling strategy, both RTS and CTA packets' sizes do not grow linearly with N. However, the tradeoff is that the packet delay would increase, since not all potential appenders can be served in each handshake.

5.7 Summary

In this chapter, we have presented a novel approach of using reverse opportunistic packet appending to improve channel utilization in underwater acoustic networks. Our idea exploits the opportunity presented in high latency scenarios, by better utilizing the idle waiting time during the 2-way RTS/CTS handshake to setup concurrent transmissions from multiple nodes. Based on this idea, we propose a sender-initiated handshaking-based MAC, called ROPA, that does not need clock synchronization. In each handshake, an initiating sender can schedule its first-hop neighbors (appenders) to transmit their appended data packets with partial overlap in time. These packet trains from different appenders will arrive at the sender (i.e., secondary data transmissions) in a collision-free manner, soon after

it finishes transmitting its own packet train to its intended receiver (i.e., primary data transmissions). This is more efficient than the conventional approach, which requires each of those neighbors to initiate a separate handshake that incurs its own overheads. Furthermore, ROPA is also equipped with a robust and versatile MAC framework that supports three possible data transmission modes: (i) both primary and secondary transmissions, (ii) only primary transmissions, and (iii) only secondary transmissions. Thus, both MACA with packet train and RIPT can be regarded as special cases of ROPA.

From our extensive single-hop and multi-hop simulations, we have shown that ROPA offers a stable saturation throughput, and provides significant gains in both throughput and delay compared to the conventional handshaking-based MAC protocols such as MACA-U, MACA-UPT, slotted FAMA, DACAP (these protocols only perform primary transmissions), as well as RIPT (it only performs secondary transmissions). We also demonstrate the importance of handling potential problematic scenarios such as the AIDC problem; the ROPA variant that does not resolve these problems, would have higher number of packet collisions as well as lower success ratio and inferior throughput. In addition, we illustrate that our adaptive mechanism, which tunes both primary and secondary train sizes, can give a performance very close to that of the optimal scheme for both throughput and delay. In the presence of packet errors, ROPA also has a better throughput efficiency than the existing MAC protocols like MACA-U and MACA-UPT.

Compared to BiC-MAC in our multi-hop simulation, we found that ROPA actually surpasses BiC-MAC in throughput and delay. Unlike BiC-MAC, ROPA does not place stringent constraints on the data packet size and inter-nodal distance, since the packet transmissions are not occur in a fully concurrent manner. Therefore, ROPA can well complement BiC-MAC for a shorter range network. While ROPA achieves better performance by exploiting partial concurrent transmissions from more number of neighbors, its MAC framework is more complex and less scalable, compared to BiC-MAC. Our study has demonstrated the value of using reverse opportunistic packet appending approach in high latency networks. Based on our findings, we contend that ROPA is an ideal protocol for ad hoc networks with limited mobility, especially in a dense network in which each node has a large number of neighbors.

Chapter 6

Saturation Throughput Analysis for Slotted BiC-MAC

6.1 Introduction

In Chapter 4, we have shown via simulation that BiC-MAC is promising for high latency networking scenario, since it can offer remarkable performance gain in terms of throughput and delay in both single-hop and multi-hop networks. As a first foray into understanding its theoretical performance, we shall focus on the saturation throughput analysis of a time slotted BiC-MAC variant in single-hop networks, under both error-free (i.e., packet loss is only due to packet collision) and error-prone (i.e., all control and data packets are subjected to certain probability of successful detection and decoding, in addition to packet collision loss) channel conditions. So far, there is lack of theoretical treatments on characterizing the throughput performance of BiC-MAC protocol. An accurate analytical model is desirable, because it not only provides a simple performance estimation for diverse operating conditions, but also is useful in optimizing the protocol's parameters. Furthermore, protocol designers can utilize the model as a more effective tool to better understand the protocol's relationship with underlying system parameters without the need of simulation, which is often time-consuming.

More importantly, the second limitation is that the existing analytical approaches from both terrestrial wireless [15, 35–39] and underwater acoustic networks [19, 30, 44, 45, 47–49], as described in Chapter 2, cannot be directly applied in analyzing BiC-MAC. Even though many terrestrial-based research works have

analyzed throughput performance of MACA [13] based MAC protocols (i.e., exchanging RTS/CTS frames prior to data packet transmission), these analytical frameworks are not applicable as they do not account for: (i) different channel characteristics such as long propagation delay and higher packet loss due to less reliable acoustic channel, and (ii) fundamentally different BiCMAC's operation rules and protocol features. We briefly highlight some essential differences from our analysis. For example, the FAMA [15] protocol's analysis of a 3-way RTS/CTS/DATA handshake do not consider neither saturation load condition nor channel errors. While the regular Markov chain analyses in [35–37] give an accurate saturation throughput for the IEEE 802.11 variants, they also do not consider channel errors and their analyzed MAC has different operation rules like backoff algorithm (freezing backoff), a unidirectional single data packet exchange, and etc. Some works also extend these analyses to account for error-prone channel [38–40]; again, the BiC-MAC analysis is quite different as we need to consider all possible frame errors from RTS/CTS/NTF, as well as each data packet in the bidirectional packet bursting.

For the throughput analysis in underwater networks, some works examine the Aloha-based protocols [44, 45, 47], while the others in [19, 30, 48, 49] study the unidirectional MACA-based protocols. Despite these analytical frameworks are proposed for underwater networks, they are also not applicable mainly because they do not account for the potential bidirectional packet exchange in high latency scenario. We now highlight the key differences against the above analyses of MACA-based protocols, since they are more relevant to our study. Earlier work in [19] analyzes a time slotted 4-way RTS/CTS/DATA/ACK handshake protocol, in a special topology where every adjacent node pairs is separated by same distance. In [30], the throughput performance of a normal unslotted 3-way RTS/CTS/DATA handshake MAC is analyzed; here, a topology of only a single receiver node and multiple transmitters is adopted. However, both of them are not saturation load analysis, and their protocol's operation rules are much simpler and backoff stage is not considered. More recent works in [48,49] attempt to derive the saturation throughput of MACA-based protocols in single-hop networks. The work in [48] directly adopts the Bianchi's [35] analytical method to analyze their unidirectional 4-way handshake MAC; however in their evaluation, they do not use the derived analytical results to verify their MAC protocol's performance.

In [49], a throughput analysis based on absorbing Markov chain is performed on a slotted RTS/CTS-based MAC with packet train, in error-free and error-prone channels (here, the probability of successful packet detection and decoding are assumed to be the same for both control and data packets). But, within a given slot, they assume that the inter-nodal propagation delay is much smaller than a single control packet's duration; while this assumption greatly simplifies the packet collision analysis, it does not allow for any bidirectional transmissions. In addition, they limit their model validation using only a small topology of 4-node.

In this chapter, we analytically study the normalized saturation throughput of slotted BiC-MAC in single-hop networks, in which upon every successful handshake, both initiating sender and its intended receiver exchange their respective batch of data packets by using a bidirectional-concurrent transmission approach. Here, the slot length is defined as a single control packet's duration plus maximum inter-nodal delay; the inter-nodal delay is also assumed to be much larger than both control and data packets' durations so as to allow for bidirectional transmissions. Our analysis on a time slotted MAC is motivated by a key insight that the time slotting mechanism losses its effects in long propagation delay environment, especially when the control packet duration is much smaller; thus, the analyzed throughput results of the slotted BiC-MAC can serve as an approximation to that of the unslotted counterpart. We propose a novel, simple analytical framework based on an absorbing Markov chain to fully model a single tagged node's operation when it attempts to transmit its batch packets; the state transition probabilities as well as expected time durations that a tagged node spent in each state are systematically derived. Note that similar to some previous works in [19, 49], the inter-nodal delay between any of the node pairs is assumed to be fixed at a maximum value, for the purpose of analytical tractability. From the resultant transition probability matrix and the fraction of time spent in each state, we can compute the average batch service time, which is then used to obtain the saturation throughput. Our Markov model not only fully captures the bidirectional packet transmission event, but also accounts for more realistic error-free and error-prone channel models, where we differentiate the probability of successful detection and decoding between control and data packets.

The remainder of this chapter is organized as follows. In Section 6.2, we first argue why a time slotting mechanism is adopted in our analytical framework,

as well as the details of slotted BiC-MAC model. Next, our general assumptions and performance metrics of interest are described in Section 6.3. In Section 6.4, we then analyze the saturation throughput of slotted BiC-MAC protocol. In Section 6.5, the accuracy of theoretical model is verified via comparison with simulation results. We also study how our analytical model can be used to closely approximate the throughputs of both slotted and unslotted BiC-MAC protocols with actual inter-nodal delays. Finally, we give our conclusion in Section 6.6.

6.2 The Slotted BiC-MAC Protocol Model

6.2.1 Motivation of Adopting a Time-Slotting Mechanism in our Analytical Framework

Although the actual BiC-MAC [69] is an unslotted protocol, we introduce a notion of time slotting into BiC-MAC and further analyze its throughput performance. Similar to other works in [19, 49], the duration of a time slot is defined as

$$t_{\text{slot}} = T_{\text{C}} + \tau_{\max}, \tag{6.1}$$

where T_{C} is the transmission time of a single control packet, and $\tau_{\max}$ is the maximum inter-nodal propagation delay. However, unlike [49], we assume that $T_{\text{C}} \ll \tau_{\max}$ so as to resemble most typical underwater network, in which the inter-nodal separation distance is usually very large. As shown in Fig. 6.1, all control packets as well as the first data packet can only be transmitted at the starting of a slot boundary.

We now explain why an analysis on this slotted BiC-MAC variant would give meaningful results. In terrestrial wireless networks, compared to an unslotted Aloha, a time slotting mechanism (suppose a slot duration is T, where T is a packet's transmission time) is useful as it can confine any packet collision to occur within a slot via synchronizing the packet transmission of all potential senders. As a result, the packet's vulnerable period (i.e., time interval over which there must not be any packet transmission; else, it will interfere with the on-going packet reception) is reduced from $2T$ in the unslotted variant to T [3]; thus, the slotting approach would considerably improve the protocol's peak throughput by a factor of two. In contrast, time slotting losses its effects in our case of $T_{\text{C}} \ll \tau_{\max}$. Even

though it synchronizes the transmission from multiple senders, more than one packet might fit within a slot without any overlapping. The occurrence of packet collision at the receiver will depend on the distribution of inter-nodal propagation delays among multiple senders, which is similar to the unslotted case. For both unslotted and slotted variants in our high latency case, we have a vulnerable period of $2T_{\mathrm{C}}$. In fact, it has been observed in [46] that the throughput of slotted protocol falls slightly below that of Pure Aloha when $a > 1$, where $a = \tau_{\max}/T$. Note that the unslotted BiC-MAC will have a slightly higher throughput than the slotted version, since the packet exchange time in each handshake is shorter, as no constraint is imposed on the transmission timing. Hence, an analysis on the slotted BiC-MAC model can be used to approximate the throughput of the unslotted variant.

6.2.2 How the Slotted BiC-MAC Protocol Works

In the actual BiC-MAC, a versatile MAC framework is designed, in which it can operate in three possible bidirectional transmission modes so as to cater for the fact that a sender-receiver (S-R) pair may not intend to exchange the same number of data packets. In this study, however, we only analyze the case when the S-R node pair exchanges the same total number of packets for each handshake. Hence, the bidirectional packet exchange is limited to two scenarios, as shown in Fig. 6.1(a)–6.1(b).

In this study, we focus on the protocol's behavior when all nodes operate at saturation traffic conditions. This will give insights into the maximum achievable throughput in a high load scenario. With this, the transmission queue of each node is assumed to be always non-empty, and is always backlogged with a batch of M number of data packets destined to any one of its first-hop neighbors. In our model, when an initiating sender transmits a batch of M packets to its intended receiver, the latter always has packet in return, and is ready to transmit M data packets as well. Similar to several existing works in [19, 49, 50], the inter-nodal delay between any of the node pairs is assumed to be fixed at $\tau_{\max}$ for the purpose of analytical tractability. While the actual inter-nodal delays are often smaller than $\tau_{\max}$ in practical setting, we will show later that our analytical results can still give a reasonably accurate estimation.

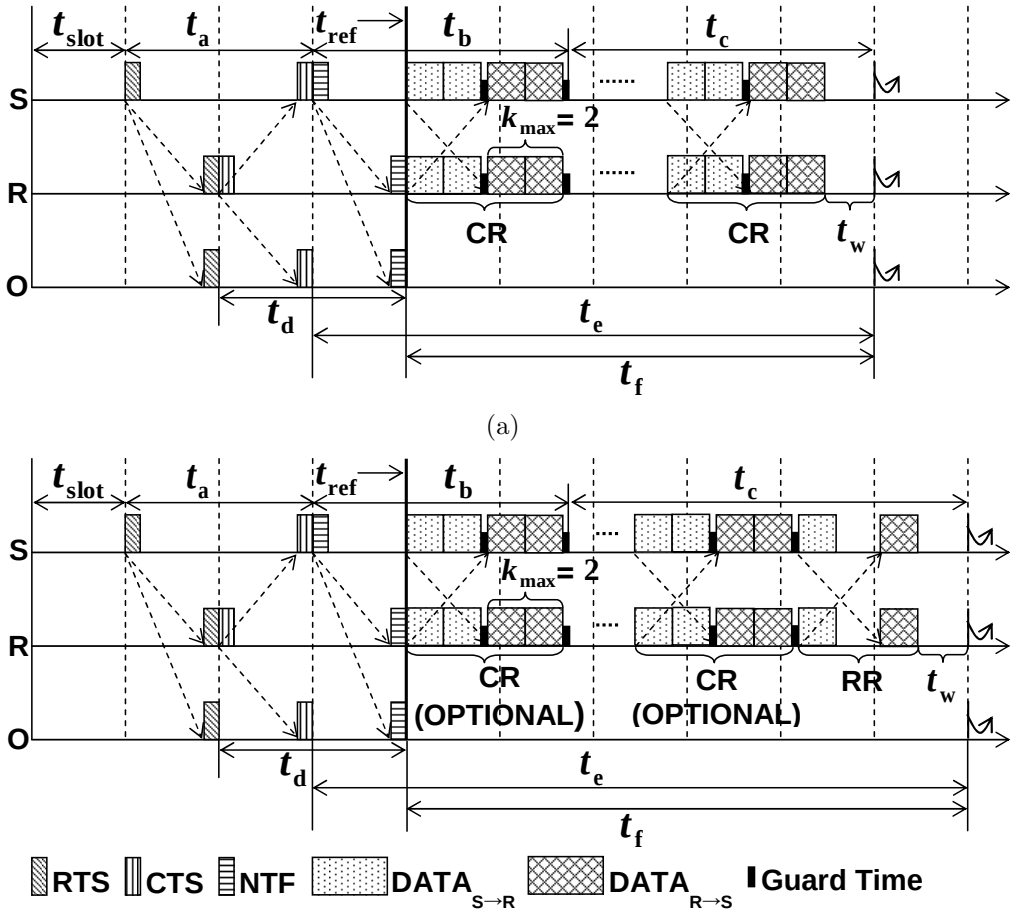


Figure 6.1: Timing diagrams of slotted BiC-MAC: (a) Type A and (b) Type B scenarios. In these two cases, both initiating sender "S" and intended receiver "R" exchange the same total number of data packets in each handshake. Node "O" is the first-hop neighbor that overhears the S-R node pair's transmissions. The releases from the current handshake are shown as curly arrows.

6.2.2.1 Channel Reservation

Fig. 6.1 shows that the two scenarios have identical sequence of control packet exchanges. Whenever a node is ready for batch transmissions, it randomly chooses any one of its neighbors as a destination, and initiates its contention timer according to the backoff interval, $t_{\rm bk}$, as:

$$t_{\rm bk} = int\_uniform(0, W-1) \times t_{\rm slot}, \tag{6.2}$$

where $int\_uniform()$ gives an integer randomly chosen from a uniform distribution over a fixed interval $[0, W-1]$.

Upon the contention timer expiry, the initiating sender will broadcast an RTS packet at the starting of a slot boundary to its first-hop neighbors. It utilizes

the RTS packet to inform its intended receiver about its total number of packets that it wishes to transmit, $n_{\mathrm{S}} = M$. Upon successfully receiving the RTS, the receiver can invite the sender for bidirectional data exchange when it has data packets in return; it responds with a CTS packet to announce the total number of packets it wishes to transmit, $n_{\mathrm{R}} = M$. In the actual BiC-MAC, when $n_{\mathrm{R}} = 0$, the handshake will reduce to a normal unidirectional RTS/CTS/data packet-train handshake. Note that the sender waits for a duration of $t_{\mathrm{a}} = 2t_{\mathrm{slot}}$ (i.e., round-trip time) for the CTS response; it will release from current handshake and start contending again when it does not fully receive the CTS.

Upon fully receiving the CTS and validating that $n_{\mathrm{R}} > 0$, the sender transmits a Notification (NTF) packet. This synchronizes the S-R pair to a common reference time, t_{ref}, so that their bidirectional packet exchange can be performed in a concurrent manner. As shown in Fig. 6.1, after transmitting an NTF, the sender waits for a duration of $\tau_{\mathrm{S,R}} = \tau_{\max}$ (inter-nodal delay of S-R pair) before it starts to transmit its packets; while the receiver can start its data transmission immediately after receiving the NTF. Notice that, the receiver also waits for a duration of $t_{\mathrm{a}} = 2t_{\mathrm{slot}}$ (timer starts once it begins its CTS transmission) for the NTF response; if it does not fully receive the NTF, it will release from handshake at t_{ref} and contend again. We also point out that from the knowledge of inter-nodal propagation delay as well as the intended number of data packets to send, both S-R nodes can locally compute a proper transmission/reception schedule for their subsequent bidirectional data exchange so as to avoid any transmit-receive (TX-RX) data collisions.

Similar to other handshaking-based MAC protocols, the expected busy duration can be carried in all control packets. Upon successfully overhearing a control packet, the neighboring nodes will remain silent by not sending any packets for the indicated busy duration, so as to avoid interfering with the on-going transmission. For example in Fig. 6.1, upon overhearing an xRTS, xCTS, and xNTF packet[1], the neighboring nodes remain silent for a respective duration of t_{a}, t_{d}, and t_{e}. They can only start a new contention upon the silent duration expiry at the slot boundary.

[1]We use "x" to denote that a control packet is destined to others.

6.2.2.2 Bidirectional-Concurrent Data Packet Transmission

As mentioned, slotted BiC-MAC can operate in 2 modes, namely, Type A and Type B scenarios when the S-R pair exchanges the same number of data packets. For a better explanation, we introduce the terms *Complete Round (CR)*, and *Residual Round (RR)*. The CR is defined as the time window when both participating nodes (i.e., S-R nodes) transmit the maximum allowable number of data packets, $k_{\max}$, bidirectionally; while RR refers to the time window when both nodes transmit bidirectionally to each other, but they transmit less than $k_{\max}$ data packets. Fig. 6.1(a)–6.1(a) show the example scenarios when $k_{\max} = 2$. In practice, small constant guard times, T_{guard}, in the order of milliseconds, can be inserted so as to accommodate for: (i) any estimation error in the inter-nodal delays, (ii) transceiver's TX-RX turnaround time, (iii) maximum sway distance caused by underwater currents, and (iv) any slight difference in the S-R pair's perception of t_{ref}. In each CR, as well as any RR that has bidirectional-concurrent transmission, the condition to avoid TX-RX collisions is simply

$$k_i T_{\text{D}} + T_{\text{guard}} \leq \tau_{\text{S,R}}, \quad i \in \{\text{S}, \text{R}\}, \tag{6.3}$$

where k_i is the number of data packets that node i transmits either in a CR or a RR. From (6.3), $k_{\max}$ can be computed as

$$k_{\max} = \lfloor (\tau_{\text{S,R}} - T_{\text{guard}})/T_{\text{D}} \rfloor. \tag{6.4}$$

The Type A scenario is characterized by the presence of at least one CR, and no RR. In contrast, the Type B scenario is characterized by the presence of optional CRs, and a single mandatory RR; thus, its transmission configuration can either be: the presence of (i) CRs and RR, or (ii) RR only. Finally, both participating nodes do not transmit any acknowledgement (ACK) upon receiving data packets. We also do not put any upper limit on the number of retries when RTS attempts fail.

6.3 System Model

6.3.1 General Assumptions

We consider an ad hoc, static, single-hop acoustic network (i.e., no hidden nodes) with finite and fixed N homogenous nodes. Each node has a single omni-directional, half-duplex underwater acoustic modem. They also exchange both control and data packets using a single channel. The transmission time of a single control and data packets are denoted as T_{C} and T_{D} seconds, respectively ($T_{\mathrm{C}} \leq T_{\mathrm{D}}$); here, the transmission time for all types of control packets is assumed to be the same, i.e., $T_x = T_{\mathrm{C}}$, where $x \in \{\mathrm{RTS}, \mathrm{CTS}, \mathrm{NTF}\}$. As mentioned, the actual BiC-MAC will revert to a unidirectional data transmission when the sender-receiver separation is too close as such $\tau_{\mathrm{S,R}} < T_{\mathrm{D}} + T_{\mathrm{guard}}$; hence, we can potentially have a mixture of unidirectional and bidirectional transmissions in the network. However, in our analysis, we focus on the case where all nodes are randomly deployed in a two-dimensional square area, but only a bidirectional transmission is allowed between any of the S-R node pairs. In order to study the protocol's robustness in the less reliable underwater channel [70], we consider an error-prone channel model, where *all* control and data packets are subjected to certain probability of successful detection and decoding. While the probability of successful packet detection depends on the nature of the packet's preamble and detection algorithm used in the physical layer, the probability of correctly decoding a packet mainly depends on how strong the forward error correction (FEC) codes is. In our model, the received data packet of duration T_{D} is subjected to a probability δ_{D} for a successful packet detection and decoding. In contrast, control packets often have stronger FEC encoding because they are critical for establishing S-R node pair handshake prior to data transmissions [71]. For simplicity, we assume that a control packet of duration T_{C} is subjected to the following probability of successful packet receptions:

$$\delta_{\mathrm{C}} = 1 - \frac{T_{\mathrm{C}}}{T_{\mathrm{D}}}(1 - \delta_{\mathrm{D}}). \tag{6.5}$$

All control packets are assumed to have the same probability, i.e., $\delta_x = \delta_{\mathrm{C}}$, where $x \in \{\mathrm{RTS}, \mathrm{CTS}, \mathrm{NTF}\}$. Note that we also consider an error-free channel in our analysis, by setting $\delta_{\mathrm{C}} = \delta_{\mathrm{D}} = 1$; here, the packet loss is solely attributed to the packet collision.

6.3.2 Performance Metrics

We are interested in the following performance metrics:

- *Batch service time*, t_{serv}*:* it is defined as an average time elapsed from the moment a batch of M data packets becomes ready for transmission at a tagged node (which includes the backoff durations), until the time at which the entire M data packets are successfully transmitted to its intended receiver plus any remaining waiting time for the next slot boundary, t_{w}, if such a duration is present. Note that when the tagged node transmits its M data packets, it also receives M number of data packets from its intended receiver, via bidirectional transmissions.

- *Saturation throughput per-node*, γ_{node}*:*

$$\gamma_{\text{node}} = \frac{\delta_{\text{D}} \cdot 2T_{\text{D}} \cdot M}{t_{\text{serv}}}. \tag{6.6}$$

  Note that each of the transmitted data packet can only be successfully received with a probability δ_{D}.

- *Saturation network throughput*, γ_{network} *(normalized):*

$$\gamma_{\text{network}} = N \cdot \gamma_{\text{node}}. \tag{6.7}$$

6.4 Saturation Throughput Analysis

From a given set of network and MAC parameters: $[N, \tau_{\max}, \delta_{\text{C}}, \delta_{\text{D}}, W, M, T_{\text{C}}, T_{\text{D}}]$, we now derive the saturation throughput of slotted BiC-MAC for single-hop networks. In Section 6.4.1, we first propose an absorbing Markov chain (AMC) to model the protocol's operation; here, we examine various time durations associated with the protocol, as well as deriving the state transition probabilities. In Section 6.4.2, we find the average batch service time, t_{serv}, based on the property of AMC; then, the throughput per-node and network throughput can be found using (6.6) and (6.7), respectively.

Table 6.1: Meaning of various states in the slotted BiC-MAC's model

State	Meaning
s_1	A tagged node starts its RTS contention cycle
s_2	After sending an RTS, the tagged node waits for CTS reply
s_3	Upon fully receiving the CTS, the tagged node transmits an NTF to its intended receiver. Then at t_{ref}, it starts its first round of bidirectional-concurrent data packet exchange
s_4	A tagged node does not transmit its RTS. It overhears an xRTS and subsequently remains silent
s_5	After successfully overhearing either a xCTS or xNTF, the tagged node extends its silent duration
s_6	A tagged node does not transmit its RTS. It fails to overhear a previously sent xRTS. But, it successfully overhears the xCTS reply, and subsequently remains silent
s_7	A tagged node does not transmit its RTS. It fails to overhear both xRTS and xCTS sent previously. But, it successfully overhears the xNTF response and subsequently remains silent
s_8	A tagged node does not transmit its RTS. There are exactly two nodes transmit their respective RTS to each other. The tagged node overhears a xCTS and subsequently remains silent
s_9	A tagged node does not transmit its RTS. There are exactly two nodes transmit their respective RTS to each other. The tagged node fails to overhear the previously sent xCTS, but it fully overhears an xNTF and subsequently remains silent
s_{10}	A tagged node receives an RTS. Then, it becomes an intended receiver for another initiating sender by replying with a CTS
s_{11}	After both CTS and NTF are successfully received, the tagged node starts its bidirectional-concurrent data exchange at t_{ref}
s_{12}	The intended receiver successfully receives the NTF, and the S-R node pair exchanges all their data packets bidirectionally

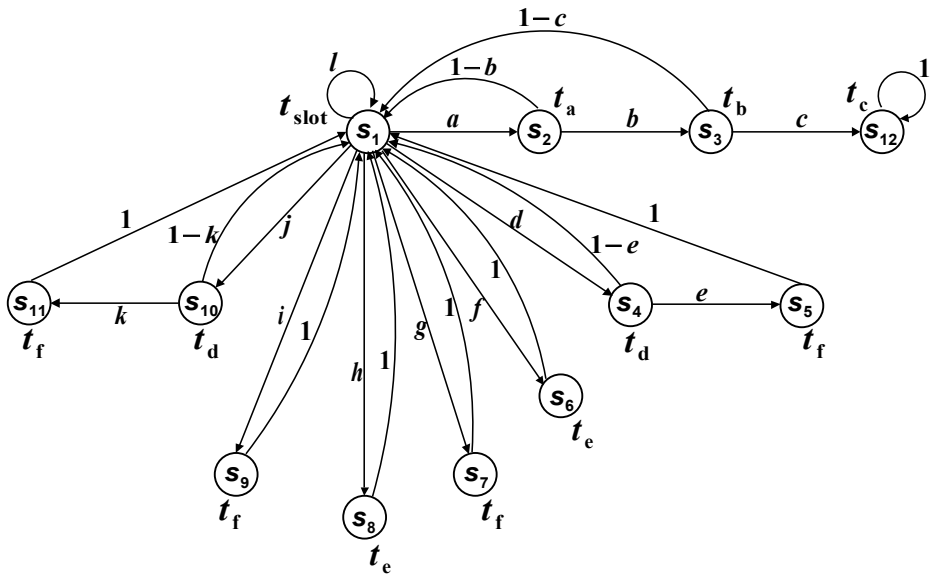


Figure 6.2: Absorbing Markov chain for modeling the operation of an arbitrary tagged node that employs the slotted BiC-MAC. The tagged node might reside in different states before it can successfully transmit its batch of data packets.

6.4.1 Modeling Slotted BiC-MAC as an Absorbing Markov Chain

Fig. 6.2 shows the AMC model for slotted BiC-MAC's batch packet transmissions. We have 12-state, $\mathcal{S} = \{s_1, s_2, \ldots, s_{12}\}$ and the meaning of each state is explained in Table 6.1; state s_{12} is an absorbing state, while the rest are transient (i.e., non-absorbing) states. The rationale of using the AMC in our analysis is due to the existence of an absorbing state in our protocol model, which results in a finite time to absorb (e.g., the batch service time). Once a batch of M data packets is successfully transmitted by a tagged node, there is a zero probability of transmitting that particular batch again. In other words, the probability of leaving the absorbing state s_{12} is zero.

6.4.1.1 Time Durations in Slotted BiC-MAC

We now explain how to compute the respective time duration that associates with each of the AMC state. They represent the time spent in a specific state, which is needed for calculating the batch service time. Note that these durations correspond to the protocol's timing requirements of both scenarios in Fig. 6.1(a)–6.1(b).

For the bidirectional transmission in each CR, both sender and receiver nodes always exchange $k_{\max}$ number of data packets (see (6.4)) with each other;

thus, the total number of CRs required for transmitting a batch of M packets is $n_{\mathrm{CR}} = \lfloor M/k_{\max} \rfloor$. Furthermore, the duration of a single CR is $t_{\mathrm{CR}} = \tau_{\max} + k_{\max} T_{\mathrm{D}} + T_{\mathrm{guard}}$. Unlike the CRs, the RR only appears in the Type B scenario; in this time window, the number of excess data packets transmitted by a sender or receiver is $k_{\mathrm{RR}} = M \bmod k_{\max}$, while the duration for the RR is simply $t_{\mathrm{RR}} = \tau_{\max} + k_{\mathrm{RR}} T_{\mathrm{D}}$.

For both scenarios in Fig. 6.1(a)–6.1(b), upon finish receiving the last bit of a batch packet exchange, there might be a waiting time, t_{w}, before a node can start a new contention at the next slot boundary. Clearly, $t_{\mathrm{w}} < t_{\mathrm{slot}}$ and it can be found as follows

$$t_{\mathrm{w}} = \begin{cases} 0, & \text{if } t^{x}_{\mathrm{over}} = 0, \\ t_{\mathrm{slot}} - t^{x}_{\mathrm{over}}, & \text{if } t^{x}_{\mathrm{over}} > 0, \end{cases} \tag{6.8}$$

where $x \in \{\mathrm{A}, \mathrm{B}\}$ scenario, $t^{\mathrm{A}}_{\mathrm{over}} = (n_{\mathrm{CR}} t_{\mathrm{CR}} - T_{\mathrm{guard}}) \bmod t_{\mathrm{slot}}$, and $t^{\mathrm{B}}_{\mathrm{over}} = (n_{\mathrm{CR}} t_{\mathrm{CR}} + t_{\mathrm{RR}}) \bmod t_{\mathrm{slot}}$.

Finally, for Type A scenario, all the time durations that are associated with the AMC can be expressed as

$$\begin{cases} t_{\mathrm{slot}} &= T_{\mathrm{C}} + \tau_{\max}, \\ t_{\mathrm{a}} &= 2t_{\mathrm{slot}}, \\ t_{\mathrm{b}} &= t_{\mathrm{slot}} + t_{\mathrm{CR}}, \\ t_{\mathrm{c}} &= (n_{\mathrm{CR}} - 1) t_{\mathrm{CR}} - T_{\mathrm{guard}} + t_{\mathrm{w}}, \\ t_{\mathrm{d}} &= 2t_{\mathrm{slot}}, \\ t_{\mathrm{e}} &= t_{\mathrm{slot}} + n_{\mathrm{CR}} t_{\mathrm{CR}} - T_{\mathrm{guard}} + t_{\mathrm{w}}, \\ t_{\mathrm{f}} &= n_{\mathrm{CR}} t_{\mathrm{CR}} - T_{\mathrm{guard}} + t_{\mathrm{w}}. \end{cases} \tag{6.9}$$

For Type B scenario, its durations are identical to (6.9), except for the time durations of t_{c}, t_{e} and t_{f}, as follows:

$$\begin{cases} t_{\mathrm{c}} &= (n_{\mathrm{CR}} - 1) t_{\mathrm{CR}} + t_{\mathrm{RR}} + t_{\mathrm{w}}, \\ t_{\mathrm{e}} &= t_{\mathrm{slot}} + n_{\mathrm{CR}} t_{\mathrm{CR}} + t_{\mathrm{RR}} + t_{\mathrm{w}}, \\ t_{\mathrm{f}} &= n_{\mathrm{CR}} t_{\mathrm{CR}} + t_{\mathrm{RR}} + t_{\mathrm{w}}. \end{cases} \tag{6.10}$$

Table 6.2: Notation used for explaining transition probabilities

Notation	Probability
y	$(N-1)\cdot 1/W\cdot(1-1/W)^{N-2}$
z	$\binom{N-1}{2}\cdot(1/W)^2\cdot(1-1/W)^{N-3}$
a	$1/W$
b	$\delta_{\mathrm{C}}^2\cdot[(1-1/W)^{N-1}+1/2\cdot y\cdot(1/(N-1))^2]$
c	δ_{C}
d	$(1-1/W)\cdot y\cdot(N-2)/(N-1)\cdot\delta_{\mathrm{C}}$
e	$\delta_{\mathrm{C}}^2+\delta_{\mathrm{C}}^3\cdot(1-\delta_{\mathrm{C}})$
f	$(1-1/W)\cdot y\cdot(N-2)/(N-1)\cdot(1-\delta_{\mathrm{C}})\cdot\delta_{\mathrm{C}}^2$
g	$(1-1/W)\cdot y\cdot(N-2)/(N-1)\cdot(1-\delta_{\mathrm{C}})^2\cdot\delta_{\mathrm{C}}^3$
h	$(1-1/W)\cdot z\cdot(1/(N-1))^2\cdot\delta_{\mathrm{C}}^2$
i	$(1-1/W)\cdot z\cdot(1/(N-1))^2\cdot\delta_{\mathrm{C}}^3\cdot(1-\delta_{\mathrm{C}})$
j	$(1-1/W)\cdot y\cdot 1/(N-1)\cdot\delta_{\mathrm{C}}+1/W\cdot 1/2\cdot y\cdot(1/(N-1))^2\cdot\delta_{\mathrm{C}}$
k	δ_{C}^2
l	$1-a-d-f-g-h-i-j$

6.4.1.2 Deriving State Transition Probabilities

Table 6.2 shows all state transition probabilities (i.e., $a, b, \ldots, l$) for our AMC. To ease our explanation, we denote $[s_x, s_y]$ as the process transits from state x to state y. Our model captures the protocol behavior from a tagged node's perspective during which it tries to send its batch packets. Specifically, before it can successfully transmit a batch packets to its intended receiver ($\{s_2, s_3, s_{12}\}$), it might be in idle backoff stage (s_1), overhearing a control packet ($\{s_4, s_5, \ldots, s_9\}$), or becoming an intended receiver for another initiating sender ($\{s_{10}, s_{11}\}$).

A backlogged tagged node starts its RTS contention cycle at state s_1. For a given slot, the probability of a node broadcasting its RTS is $a = 1/W$. Thereafter, the initiating sender transits to state s_2 and waits for the CTS reply. If the sender successfully receives its CTS, it will transmit an NTF and start its first round of bidirectional-concurrent data exchange at t_{ref} in state s_3; else, it releases from the current handshake and starts contending again in state s_1. There are two mutually exclusive events that yield a transition $[s_2, s_3]$. For the first event (e_1), only the tagged sender is allowed to transmit its RTS in a given slot (remaining $N-1$ neighbors cannot transmit and thus no RTS collision), as well as both RTS and CTS (sent by the intended receiver after it receives the RTS) packets

must be successfully received at the intended receiver and initiating tagged sender, respectively. So, $p\{e_1\} = (1 - 1/W)^{N-1} \cdot \delta_{\mathrm{C}}^2$, where δ_{C}^2 is due to both RTS and CTS must be detected and decoded, independently. For the second event (e_2), when $T_{\mathrm{C}} \ll \tau_{\max}$, there is no RTS collision if both tagged sender and its intended receiver transmit their RTS to each other in a same slot. Only one of them will reply a CTS so that the handshake can proceed normally, as in Fig. 6.1. The deadlock can be resolved by comparing whose unique ID is larger; thus, the tagged sender has a probability 0.5 for receiving its CTS. Moreover, both RTS and CTS must be fully received, similar to e_1. The probability of exactly a single node out of $N - 1$ nodes transmits in the slot is $y = \binom{N-1}{1} \cdot 1/W \cdot (1 - 1/W)^{N-2}$; while the probability of both nods intend to send to each other is $(1/(N-1))^2$. So, $p\{e_2\} = 1/2 \cdot y \cdot (1/(N-1))^2 \cdot \delta_{\mathrm{C}}^2$. Finally, we have $b = p\{e_1\} + p\{e_2\} = \delta_{\mathrm{C}}^2 \cdot [(1-1/W)^{N-1} + 1/2 \cdot y \cdot (1/(N-1))^2]$. In s_3, the tagged sender also waits for incoming data packets. Those packets can only be sent if its intended receiver has fully received the NTF; thus, a transition $[s_3, s_{12}]$ occurs with probability $c = \delta_C$ and the S-R pair exchanges all their packets bidirectionally. The tagged sender releases from the current handshake if the NTF is lost (transition $[s_3, s_1]$).

When a tagged node does not send its RTS in a given slot (probability $1 - 1/W$), it decreases its backoff timer by one slot duration. During that slot, if exactly a single node (say node A) from $N-1$ nodes sends an RTS (probability y) that destined to anyone of its neighbors, except for the tagged node (probability $(N-2)/(N-1)$, as a node would not pick itself as a destination). Then, a transition $[s_1, s_4]$ occurs when the tagged node fully overhears the broadcasted xRTS (probability δ_{C}) and remains silent, so as to protect CTS and NTF receptions at the sender and its intended receiver, respectively. Therefore, we have $d = (1 - 1/W) \cdot y \cdot (N-2)/(N-1) \cdot \delta_{\mathrm{C}}$. Next, to avoid interfering with S-R pair's data exchange, the tagged node's silent duration must be extended via overhearing either a xCTS or xNTF. Here, the probability for it to fully overhear a xCTS is δ_{C}^2, because: (i) node A's RTS must be received at its intended receiver (probability δ_{C}) so that a CTS can be sent, and (ii) the xCTS must be overheard at the tagged node (probability δ_{C}). Although it might miss the xCTS (probability $\delta_{\mathrm{C}} \cdot (1 - \delta_{\mathrm{C}})$), it could still overhear the xNTF to extend its silent duration (probability δ_{C}^2, as both CTS and xNTF must be received correctly at node A and tagged node, respectively). For these events, a transition $[s_4, s_5]$ occurs with probability $e =$

$\delta_{\mathrm{C}}^2 + \delta_{\mathrm{C}}^3 \cdot (1 - \delta_{\mathrm{C}})$; else, it releases from the handshake (transition $[s_4, s_1]$). Upon its silent period expiry in state s_5, the tagged node transits to s_1 and contends again.

We have dealt with the case of a tagged node fully overhears a xRTS, as one other node (say node A) transmits. Now, we focus on the case where it misses the xRTS, but still overhears either a subsequent xCTS or xNTF and remains silent. The probability of an idle tagged node misses a xRTS, is $(1 - 1/W) \cdot y \cdot (N-2)/(N-1) \cdot \delta_{\mathrm{C}}$. Meanwhile, node A's intended receiver only sends a CTS upon receiving a RTS (probability δ_{C}); then, a transition $[s_1, s_6]$ occurs when this CTS reply is fully overheard at the tagged node (probability δ_{C}). So, $f = (1-1/W) \cdot y \cdot (N-2)/(N-1) \cdot (1-\delta_{\mathrm{C}}) \cdot \delta_{\mathrm{C}}^2$. Next, we discuss the case where the tagged node misses both xRTS and xCTS, but successfully overhears a xNTF (i.e., transition $[s_1, s_7]$). Here, the probability of an idle tagged node misses both xRTS and xCTS, is $(1-1/W) \cdot y \cdot (N-2)/(N-1) \cdot (1-\delta_{\mathrm{C}})^2 \cdot \delta_{\mathrm{C}}$; the term "$(1-\delta_{\mathrm{C}})^2 \cdot \delta_{\mathrm{C}}$" is because: (i) it misses the xRTS (probability $1-\delta_{\mathrm{C}}$), (ii) node A's intended receiver receives the RTS (probability δ_{C}) and sends a CTS, and (iii) it misses the xCTS (probability $1 - \delta_{\mathrm{C}}$). Thereafter, the probability that the tagged node overhears a xNTF correctly is δ_{C}^2, as CTS and xNTF must be fully received at node A and tagged node, respectively; the CTS must be properly received at node A before it can send an NTF. So, $g = (1 - 1/W) \cdot y \cdot (N-2)/(N-1) \cdot (1-\delta_{\mathrm{C}})^2 \cdot \delta_{\mathrm{C}}^3$. Upon the silent period expiry in s_6 and s_7, the tagged node returns to s_1.

So far, we address the case of one other node sends in a slot, while a tagged node overhears the transmission. We now deal with the case where exactly two other nodes send their respective RTS to each other, which can still yield a bidirectional data exchange. Recall that only one of them will reply with a CTS. Here, a tagged node will remain silent upon overhearing either a xCTS (transition $[s_1, s_8]$) or xNTF (transition $[s_1, s_9]$). Note that it will not overhear a xRTS, as both RTS packets arrive and overlap at the same time instant due to the $\tau_{\max}$ assumption. The probability of a tagged node overhearing a xCTS when two other nodes send their RTS to each other, is $h = (1 - 1/W) \cdot z \cdot (1/(N-1))^2 \cdot \delta_{\mathrm{C}}^2$, where we define $z = \binom{N-1}{2} \cdot (1/W)^2 \cdot (1-1/W)^{N-3}$; the term "$\delta_{\mathrm{C}}^2$" is because both RTS and xCTS must be received correctly at the intended receiver and tagged node, respectively. For the transition $[s_1, s_9]$, the tagged node misses the xCTS, but it still overhears a xNTF later and remains silent. The probability of this event is

$i = (1 - 1/W) \cdot z \cdot (1/(N-1))^2 \cdot \delta_C^3 \cdot (1 - \delta_C)$; we have "$\delta_C^3 \cdot (1 - \delta_C)$" because: (i) an RTS is properly received at the intended receiver, but the tagged node fails to overhear the receiver's CTS (probability $\delta_C \cdot (1 - \delta_C)$), and (ii) the CTS reply is correctly received at the initiating sender, and the tagged node overhears the sender's xNTF (probability δ_C^2). For these cases, the tagged node goes to s_1 upon its silent duration expiry.

We now examine the case where a tagged node becomes an intended receiver node (i.e., transition $[s_1, s_{10}]$, where it sends a CTS upon receiving an RTS) for another initiating sender. There are two possible events that lead to this case. First, a tagged node does not transmit in a given slot, while exactly one other node from $N - 1$ neighbors sends an RTS that destined to it; then, it will reply a CTS upon fully receiving the RTS. The probability of this event is $(1 - 1/W) \cdot y \cdot 1/(N-1) \cdot \delta_C$, following our earlier argument. Second, there is exactly one other neighbor (say node A) sends an RTS to a tagged node in a slot, while the latter also sends its RTS destined to node A in the same slot; the tagged node has a probability of 0.5 to reply a CTS upon fully received node A's RTS. The probability of this event is $1/W \cdot y \cdot (1/(N-1))^2 \cdot 1/2 \cdot \delta_C$. Thus, $j = (1 - 1/W) \cdot y \cdot 1/(N-1) \cdot \delta_C + 1/W \cdot y \cdot (1/(N-1))^2 \cdot 1/2 \cdot \delta_C$. Upon sending its CTS, the tagged receiver waits for an NTF from node A in state s_{10}. It transits to state s_{11} if it successfully receives the NTF; else, it releases from the handshake and goes to s_1. The transition $[s_{10}, s_{11}]$ occurs with probability $k = \delta_C^2$, since both CTS and NTF must be fully received at node A and tagged node, respectively. Then in s_{11}, the tagged node can start its bidirectional data exchange at t_{ref}. Lastly, a self-transition $[s_1, s_1]$ occurs when none of the nodes are sending, or either an RTS, CTS, or NTF is not properly received by a tagged node as discussed above. From the AMC, this self-transition probability is $l = 1 - a - d - f - g - h - i - j$.

6.4.2 Saturation Throughput of Slotted BiC-MAC

For our 12-state AMC in Fig. 6.2, its transition probability matrix $\mathbf{P}$ can be expressed as:

$$\mathbf{P} = \left[\begin{array}{cccccccccccc|c} l & a & 0 & d & 0 & f & g & h & i & j & 0 & 0 \\ 1-b & 0 & b & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1-c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c \\ 1-e & 0 & 0 & 0 & e & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1-k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right], \tag{6.11}$$

where the (i, j)-entry denoted by $p_{i,j}$ is the probability of transition from state i to state j. Note that this matrix has a canonical form [72] of

$$\mathbf{P} = \left[\begin{array}{c|c} \mathbf{Q} & \mathbf{R} \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right], \tag{6.12}$$

where $\mathbf{I}$ is a 1×1 identity matrix, $\mathbf{Q}$ is a 11×11 matrix that contains the transition probabilities among transient states, $\mathbf{R}$ is a 11×1 matrix that defines the transition probabilities from the transient states to the absorbing state s_{12}, and $\mathbf{0}$ is a 1×11 zero matrix. From the AMC's property, the fundamental matrix for matrix $\mathbf{P}$ is

$$\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1}, \tag{6.13}$$

where the entry $n_{i,j}$ of $\mathbf{N}$ gives the expected number of times that the process visits transient state s_j, given that it started in transient state s_i.

In order to find the slotted BiC-MAC's saturation throughput, we need to

compute a tagged node's batch service time, t_{serv}. This duration is equivalent to the average time taken for the Markov process to be fully absorbed. As mentioned, before a tagged node can successfully transmit its batch data packets, it might reside in any of the following stages: (i) performing backoff procedure, (ii) overhearing control packets and remain silent, and (iii) becoming an intended receiver for another initiating sender. Therefore, t_{serv} can be expressed as

$$t_{\text{serv}} = t_{\text{trans}} + t_{\text{abs}}, \tag{6.14}$$

where t_{trans} is the total amount of time spent in each of the transient state *before* the Markov process being absorbed, given that a node initially started in state s_1; while t_{abs} is the time spent by a node when it resides in the absorbing state.

To find t_{trans}, we first define an initial probability vector $\mathbf{u} = [1\ 0\ 0\ \ldots\ 0]$ (size 1×11), as a backlogged tagged node always starts its RTS contention cycle in s_1. We also define a time vector $\mathbf{t} = [t_1\ t_2\ \ldots\ t_{11}]^T$, where t_x is the time spent by a node when it resides in transient state x; for example in Fig. 6.2, $t_1 = t_{\text{slot}}, t_2 = t_{\text{a}}$, and etc. Then, t_{trans} can be found as

$$t_{\text{trans}} = \mathbf{uNt} = \sum_{j=1}^{11} (n_{1,j} \cdot t_j). \tag{6.15}$$

Note that the expression for $n_{1,j}$ can be found in the Appendix. Finally, $t_{\text{abs}} = t_{\text{c}}$ simply because the process only visits the absorbing state s_{12} once, and spends t_{c} duration to exchange the remaining packets bidirectionally. Upon obtaining t_{serv}, we can compute the saturation throughput using (6.6) and (6.7).

6.5 Performance Evaluation

6.5.1 Simulation Model

We have developed a custom C++ discrete event-driven network simulator to evaluate our analytical model. For the network topology, we consider an ad hoc single-hop acoustic network with $N = 4$ and 50 static nodes, where all nodes are within each other's range. Each node is equipped with a half-duplex omni-directional transceiver. All nodes are randomly deployed in a square area of 5000×5000 m^2, but the resultant inter-nodal delay between any of the S-R pairs only

allows the pair to exchange data packets in a bidirectional transmission manner. For the 4 and 50-node topologies, their maximum inter-nodal delays $\tau_{\max}$ are 3.1127 s and 4.5715 s, respectively. For the channel model, we consider an error-free (i.e., the packet loss is solely due to packet collision), as well as an error-prone ($\delta_{\mathrm{C}} < 1, \delta_{\mathrm{D}} < 1$) channel so as to study the protocol's robustness in a more realistic scenario. For a given δ_{D}, we use (6.5) to compute δ_{C}. As for the MAC parameters, we set $W = N + 1$, $T_{\mathrm{guard}} = 10$ ms, $T_{\mathrm{C}} = 0.05$ s, and $T_{\mathrm{D}} = 0.38$ s. Note that the aforementioned settings resemble most typical operating scenario for underwater acoustic network. We also do not put any upper limit on the number of retries when RTS attempts fail. To study the saturation performance, each node is assumed to be always backlogged with a batch of M number of data packets to be sent to any one of its neighbors. Whenever a node is ready to initiate a new handshake, it will randomly pick any one of its neighbors as a destination with equal probability. Finally, each simulation point runs for 1×10^6 s, before we collect the simulation results.

In order to validate our analytical model, as well as comparing with the BiC-MAC protocols in a more practical setting, we simulate the following variants of BiC-MAC.

- *Slotted BiC-MAC with fixed maximum inter-nodal delay:* The operation of this variant follows our analytical BiC-MAC model. Note that the inter-nodal delay between any of the node pairs is artificially set to $\tau_{\max}$, where its value is determined from the maximum possible inter-nodal delay among all node pairs in an actual topology.

- *Slotted BiC-MAC with actual inter-nodal delay:* Instead of the $\tau_{\max}$ assumption, we follow realistic inter-nodal delays based on the actual node deployment. This is still a slotted MAC protocol; thus, all control packets, as well as the first data packet in a batch will be transmitted at the starting of a slot boundary.

- *Unslotted BiC-MAC with actual inter-nodal delay:* We do not adopt any slotting mechanism and the packet transmission can start anytime. Its backoff interval's computation is similar to (6.2), except that the integer restriction is not imposed. Since there is no constraint on the packet transmission's timing, an S-R pair's channel reservation and data exchange can

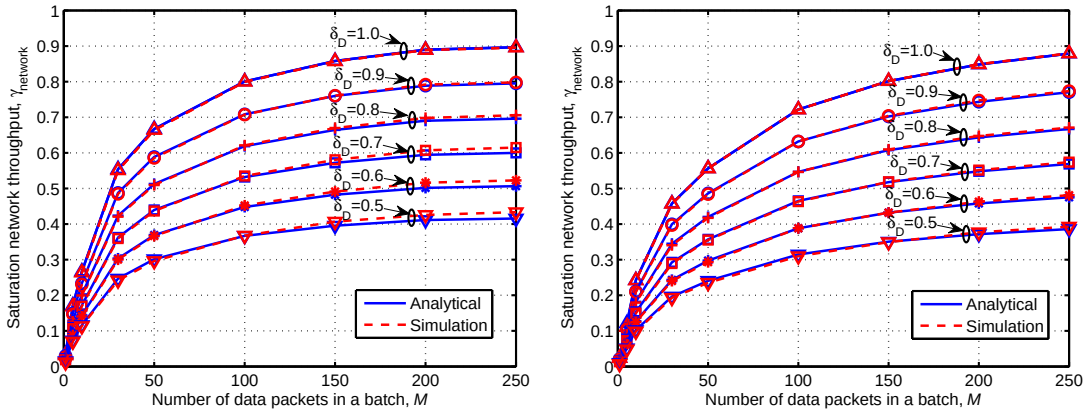


Figure 6.3: Verification of the slotted BiC-MAC's analytical model by comparing against simulation results: (a) 4-node and (b) 50-node scenarios. Here, the probability δ_D is varied from 0.5 to 1.0. Note that all inter-nodal delays are set to τ_{max} in the slotted BiC-MAC's simulation.

be accomplished in a much shorter duration. The overhearing nodes also have a shorter silent duration. Finally, the inter-nodal delay between any of the node pairs is based on the actual node deployment.

6.5.2 Numerical and Simulation Results

6.5.2.1 Verification of Analytical Model

We validate our analytical model by comparing against simulation results. For the slotted BiC-MAC's simulation, all inter-nodal propagation delays are set to τ_{max}, so that this matches our analytical model's assumption. We examine the slotted BiC-MAC's saturation network throughput for both 4-node (Fig. 6.3(a)) and 50-node (Fig. 6.3(b)) cases, as the probability δ_D is varied from 0.5 to 1.0. As can be seen, the analytical results agree very well with the simulation results; thus confirming the accuracy of our analysis approach. Note that we have also evaluated for other network sizes as well as different packet durations, and reaches similar conclusion. These results are useful for network with special geometry where any adjacent two nodes is separated by same distance, such as an equilateral triangle, or regular tetrahedron node arrangement for 3D-underwater networks.

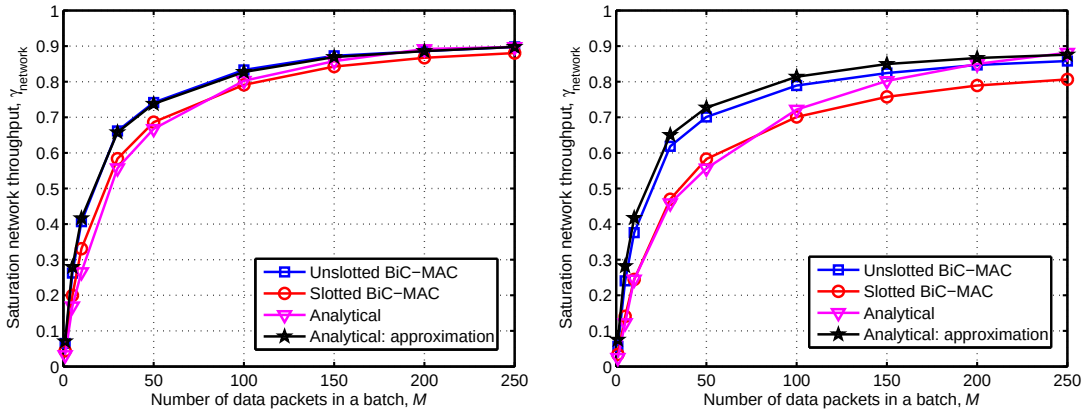


Figure 6.4: Approximating throughputs of both slotted and unslotted BiC-MAC with actual inter-nodal delays, by using the analytical result (which assumes $\tau_{\max}$). We also compare with another approximation approach that utilizes the information of actual inter-nodal delays in the analytical expression. Here, we set probability $\delta_{\mathrm{D}} = 1.0$: (a) 4-node and (b) 50-node scenarios.

6.5.2.2 Comparison of Analytical Results against BiC-MAC Protocols with Actual Inter-nodal Delay

We compare our analytical results against the simulation results of both slotted and unslotted BiC-MAC's in a more practical setup, where all node pairs have actual inter-nodal delays, instead of $\tau_{\max}$. Fig. 6.4(a)–6.4(b) show the respective saturation throughput comparisons for the 4 and 50-node cases, as the probability $\delta_{\mathrm{D}} = 1.0$.

As explained, while the use of time slot *drastically* improves the peak normalized throughput of terrestrial's unslotted Aloha from about 18.4% to 36.6%, this approach losses its effects when $T_{\mathrm{C}} \ll \tau_{\max}$. Thus, we do not expect an enormous performance difference between the unslotted and slotted BiC-MAC. As shown, the unslotted BiC-MAC consistently outperforms the slotted one for all batch sizes M; as an example when $M = 100$, the unslotted BiC-MAC's throughputs for $N = 4$ and $N = 50$ nodes, are 5.21% and 12.42%, respectively, better than that of the slotted case. This is expected because no timing constraint is imposed on the unslotted BiC-MAC; thus, an unslotted tagged node has shorter: (i) backoff interval, (ii) data exchange time between S-R pair in each handshake, and (iii) silent period for overhearing nodes. The reason why the throughput gap increases as N grows is explained as follows. We have set the uniform backoff window $W = N + 1$; so, a tagged node in 50-node case is less likely to transmit its RTS in a given slot (probability 1/51), compared to the 4-node case (probability

1/5). As N grows, a tagged node spends a larger proportion of time in activities such as overhearing others' transmissions (and remain silent) or becoming an intended receiver node, before being able to initiate its own batch transmission. As a result, the throughput gap of unslotted and slotted variants becomes more evident for a larger N. Note that the batch service time also increases as N grows.

We now compare the analytical result (which assumes $\tau_{\max}$; labeled as "Analytical" in Fig. 6.4(a)–6.4(b)) against the slotted BiC-MAC's simulation result. For both $N = 4$ and 50 cases, the analytical results fall marginally below that of the slotted BiC-MAC when M is small; however as M is further increased, the analytical model overestimates its throughput performance. This can be explained using two opposing factors. First, the slotted BiC-MAC could have a higher throughput, as no RTS collision is likely to occur when multiple senders transmit their RTS simultaneously in a slot (due to the different inter-nodal delays in actual topology); thus the handshake can proceed normally. In contrast, we have a certain RTS collision event in the analytical model, due to the assumption of $\tau_{\max}$ (except when exactly two senders transmit their RTS to each other). Second, the analytical model only has little time wastage Δ in each CR, where $\Delta = [(\tau_{\max} - T_{\text{guard}}) \bmod T_{\text{D}}] \times T_{\text{D}}$ and $0 \leq \Delta < T_{\text{D}}$; it is a "gap" that appears after a node sends its last packet within a CR, but before it starts to receive the incoming packet. This time window is needed to avoid any TX-RX collision. From our parameters in Section 6.5.1, Δ for the 4 and 50-node cases are found to be 0.0627 s and 0.0015 s, respectively, which is a very small loss. In contrast, the slotted BiC-MAC would have a larger time wastage in each CR, due to the inter-nodal delays from an actual topology are less likely to give a tightly packed CR. Thus, the slotted BiC-MAC has more wastage as M increases, and its throughput performance falls below that of the analytical estimation.

For a small network (e.g., Fig. 6.4(a)), the analytical model closely approximates the slotted BiC-MAC's throughput performance; also, it is useful in predicting the unslotted BiC-MAC's throughput, since throughput gap between the unslotted and slotted protocols is quite small. For a large network (e.g., Fig. 6.4(b)), the throughput gap becomes progressively wider due to the aforementioned reason. While the analytical model still matches the slotted BiC-MAC's throughput reasonably well, especially when $M \leq 100$, it becomes less accurate in estimating the unslotted BiC-MAC's throughput.

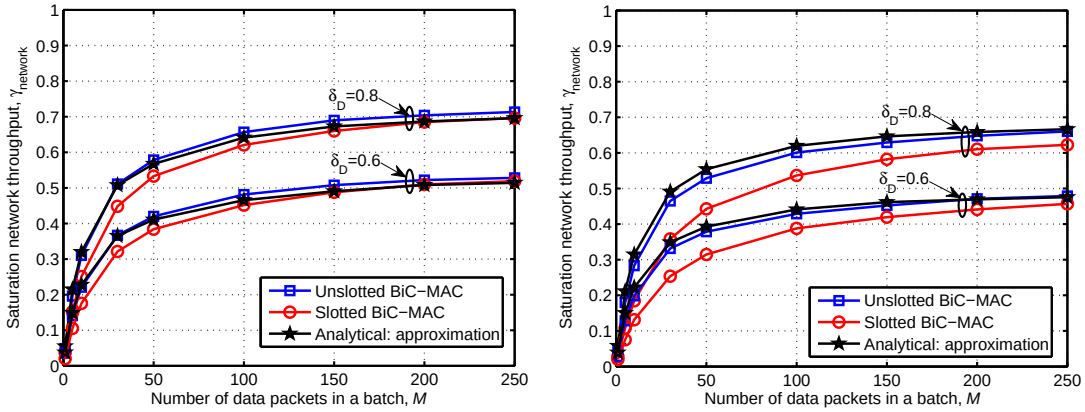


Figure 6.5: Approximating throughputs of both slotted and unslotted BiC-MAC with actual inter-nodal delays, by using an approximation approach that uses the information of actual inter-nodal delays in the analytical expression. Here, we set probability $\delta_{\mathrm{D}} = \{0.6, 0.8\}$: (a) 4-node and (b) 50-node scenarios.

In actual network topology, the inter-nodal propagation delays are often smaller than $\tau_{\max}$. In an attempt to have a better throughput approximation, we propose a new approximation approach that utilizes the information of actual inter-nodal delays in our analytical expression. Specifically, in a single-hop network of N nodes, the total number of communication links between any of the node pairs (D), is

$$D = \sum_{i=1}^{N-1}(N-i) = \frac{N(N-1)}{2}. \tag{6.16}$$

All possible inter-nodal delays for the network can be expressed as $[\tau_1\ \tau_2\ \ldots\ \tau_{\mathrm{D}}]$. Using each τ_i, the approximated network throughput $\hat{\gamma}_{\mathrm{network}}$, can be found as

$$\hat{\gamma}_{\mathrm{network}} = \left[\sum_{i=1}^{D}\gamma_{\mathrm{network}}(\tau_i)\right] \Big/ D, \tag{6.17}$$

where we substitute $\tau_{\max}$ by τ_i in the analytical expression and evaluate the resultant throughput due to τ_i, before averaging all resultant throughputs over D. The rationale behind is that, the throughput approximation using $\tau_{\max}$ alone may lead to overestimation or underestimation, which depends on the parameters chosen such as packet length and topology size; in this case, (6.17) can average out these effects.

Fig. 6.4(a)–6.4(b) show the results of $\hat{\gamma}_{\mathrm{network}}$ (i.e., "Analytical: approximation"). Interestingly, we found that the approximation is very close to the

throughput performance of unslotted BiC-MAC for both $N = 4$ and 50. This is particularly useful to complement the less accurate throughput approximation by using $\tau_{\max}$ alone, especially when N is large. We also show $\hat{\gamma}_{\text{network}}$ in Fig. 6.5(a)–6.5(b) for error-prone channel ($\delta_{\text{D}} = \{0.6, 0.8\}$); it can be seen that the approximated throughputs are also very close to that of the unslotted BiC-MAC.

6.6 Summary

In this chapter, we have analytically studied the normalized saturation throughput of slotted BiC-MAC in single-hop networks, under error-free and error-prone channel models. While the original BiC-MAC is an unslotted MAC, our rationale of analyzing a slotted MAC is based on a key insight that a time slotting mechanism losses its effects in our case when the inter-nodal propagation delay is much longer than a single control or data packet's transmission time (this condition allows for bidirectional transmissions). Therefore, the slotted analytical results can serve as an approximation to the throughput performance of unslotted counterpart, since their performance would not be too far apart.

We have proposed a novel, simple analytical framework based on an absorbing Markov chain to model the protocol behavior of a single tagged node as it attempts to exchange its backlogged batch of data packets, by using a bidirectional-concurrent transmission approach. The state transition probabilities as well as expected time durations that a tagged node spent in each state are thoroughly derived. By utilizing the resultant transition probability matrix and the fraction of time spent in each state, we can compute the average batch service time, which is then used to obtain the saturation throughput. Compared to the existing analytical models for underwater networks, the unique features of our analytical framework are: (i) a more systematic modeling of a tagged node's protocol operation, (ii) the possibility of bidirectional-concurrent packet exchange is completely captured, which significantly differs from the conventional unidirectional transmission approach, and (iii) a more realistic error-free and error-prone channel models are considered.

From comparisons against the simulated slotted BiC-MAC (all inter-nodal delays are fixed at a maximum value, so as to match our model's assumption) in small and large networks, we have verified that our proposed Markov model can

give extremely accurate normalized saturation throughput results for both error-free and error-prone channel conditions. To show the usefulness of our analytical model, we also compare its results against the simulation results of both unslotted and slotted BiC-MAC protocols with actual inter-nodal delays. For both network sizes, we have shown that our analytical throughput results can approximate the throughput performance of slotted BiC-MAC reasonably well. The analytical result, in turn, can also be used to estimate the unslotted BiC-MAC's performance, since the throughput gap between unslotted and slotted variants are quite close, especially when the number of nodes in network is small. However, we observed that this throughput gap will be widened when the number of nodes is large, and it becomes less accurate for estimating the unslotted BiC-MAC's throughput.

Finally, we also propose another approximation approach that utilizes the information of actual inter-nodal delays in the analytical expression. From both simulated network topologies, we found that it can closely approximate the saturation throughput of unslotted BiC-MAC with actual inter-nodal delays, for error-free and error-prone scenarios. This is particularly useful to complement our previous throughput approximation method (that assumes maximum inter-nodal delays), which becomes less accurate in estimating the unslotted MAC performance when the number of node is large. Nonetheless, we point out that the number of communication node in underwater networks are usually small due to much expensive device, compared to terrestrial wireless nodes.

The significance of our analytical model are: (i) it allows protocol designers to quickly estimate both unslotted and slotted BiC-MAC protocols' throughput performance, without the need of simulation that is often time-consuming, (ii) it is useful for optimizing system parameters such as backoff window, batch size, packet length, and etc.

Chapter 7

The MAT-Normalized Throughput Metric

7.1 Introduction

By exploiting spatial reuse in multi-hop settings, nodes sufficiently far apart can transmit simultaneously to improve overall network performance. To this end, there are many works that focus on medium access control (MAC) protocol design and its performance evaluation [19, 73–81]. Rather than proposing yet another protocol, we explore the use of throughput metrics in evaluating MAC protocols in static multi-hop wireless networks. In this chapter, we shall primarily focus on multi-hop networks with negligible propagation delay, before extending our proposed throughput metric for evaluating BiC-MAC and ROPA protocols in a long propagation delay environment.

Throughput in single-hop networks is well understood [81]. The computed throughput in bits per second (bps) can be normalized with the link transmission rate R (assuming that all nodes use the same rate), so that the resultant value is within the range $[0, 1]$. This is essentially benchmarking against the maximum achievable throughput since only a single node can successfully deliver its data packets at rate R, in a network with negligible propagation delay. Unfortunately, this is not so straightforward for multi-hop networks. From the literature, two commonly adopted throughput metrics for multi-hop networks are: *aggregate throughput* (also called network/system throughput) [73–75] and *throughput per-node* (also called per-station/per-user throughput) [76–78]. Aggregate throughput is the summation of the throughputs of all nodes in a network. For throughput

per-node, the aggregate throughput is divided by the total number of nodes. Some works also normalize throughput per-node by link rate R [19, 80, 81]; this is called *rate-normalized throughput per-node*.

We now explain two limitations that arise from the use of the aforementioned multi-hop throughput metrics. First, these metrics are not a generalization of the single-hop throughput metric, in that they do not seek to benchmark against the maximum achievable throughput. Note that the maximum achievable throughput in multi-hop networks is different from the single-hop case. Here, the quantity we are interested in is the maximum *aggregate* data rate that can be supported for all nodes simultaneously in a given network topology, rather than only a single node's rate R. Hence, the existing multi-hop throughput metrics do not provide as much intuition as the single-hop throughput metric, with regard to the performance relative to the best achievable. For instance, the use of rate-normalized throughput per-node often results in a very low normalized value (e.g., on the order of 10^{-2} in [19, 80, 81]), and does not provide any hint about how far it is from the best achievable. As another example, we will show later that when Aloha is applied in a multi-hop network with string topology, the peak throughput is actually quite close to the theoretical peak for single-hop networks, which is around 18% of the maximum achievable throughput. This cannot be appreciated if the rate-normalized throughput per-node metric were used, as it only gives a normalized, unitless value of around 0.09.

Second, many previous works only compare a proposed MAC protocol's throughput against that of a de facto MAC, such as the IEEE 802.11 [73–75]. This comparison approach is inadequate for performance analysis, because it only portrays a *relative* performance improvement/degradation. Instead, a better approach would be benchmarking with respect to the best achievable bit-rate, which gives an absolute performance measure, and is often of greater interest to protocol designers.

Here, we propose a unified normalized throughput metric, which allows the existing normalized throughput of both single and multi-hops to be expressed in a general formula. Since the current multi-hop metrics do not yield much insight on the best achievable bit-rate, we present a new variant, that benchmarks against the Maximum Achievable Throughput (MAT). The MAT-normalized throughput is characterized by the product of maximum number of successful simultaneous

transmissions and link rate. To compute the former, a binary integer linear programming (BILP) problem is formulated. We next demonstrate the use of our metric in both string and square grid topologies. We also derive exact mathematical expressions for the maximum successful simultaneous transmissions for these two topologies. Unlike existing metrics, our metric allows for better performance comparison across different MAC protocols. Finally, we extend the use of MAT-normalized throughput metric to evaluate BiC-MAC and ROPA in the string and square grid networks, under the presence of long propagation delay.

The remainder of this chapter is organized as follows. We explain our proposed throughput metric in Section 7.2. Then, we demonstrate its use in Section 7.3. In Section 7.4, we extend our metric to evaluate BiC-MAC, ROPA and MACA-U protocols under long propagation delay setting. Lastly, a conclusion is given in Section 7.5.

7.2 Our Proposed Throughput Metric

7.2.1 The Unified Normalized Throughput Metric

We first summarize three existing throughput metrics that are commonly used for evaluating a given MAC protocol $\mathcal{P}$:

- Aggregate throughput (in bps), $\gamma_{\text{ag}}(\mathcal{P})$:

$$\gamma_{\text{ag}}(\mathcal{P}) \triangleq \frac{\sum_{i=1}^{i=n} r_i(\mathcal{P}) \cdot L_{\text{DATA}}}{T}, \tag{7.1}$$

  where n is the total number of nodes, L_{DATA} is the data packet's payload length in bits, $r_i(\mathcal{P})$ is the total number of data packets successfully received by destination i in a duration of T seconds; the total number of data packets received depends on the MAC protocol employed.

- Throughput per-node (in bps), $\gamma_{\text{nd}}(\mathcal{P})$:

$$\gamma_{\text{nd}}(\mathcal{P}) \triangleq \gamma_{\text{ag}}(\mathcal{P})/n. \tag{7.2}$$

- Rate-normalized throughput per-node (unitless), $\gamma_{\text{r}}(\mathcal{P})$:

$$\gamma_{\text{r}}(\mathcal{P}) \triangleq \gamma_{\text{nd}}(\mathcal{P})/R = \gamma_{\text{ag}}(\mathcal{P})/[n \cdot R]. \tag{7.3}$$

To complement the existing throughput metrics, we propose a unified normalized throughput metric, $\gamma_{\text{norm}}(\mathcal{P})$:

$$\gamma_{\text{norm}}(\mathcal{P}) \triangleq \gamma_{\text{ag}}(\mathcal{P})/[\beta \cdot R], \tag{7.4}$$

where $\beta > 0$ is a normalization factor. For single-hop networks, we set $\beta = 1$ and (7.4) reduces to the usual normalized throughput metric. For multi-hop networks, $\beta = 1/R$, $\beta = n/R$ and $\beta = n$ give (7.1), (7.2) and (7.3) respectively. Unlike the single-hop's throughput notion, these existing variants of the multi-hop throughput metric do not seek to normalize by the best achievable bit-rate. We therefore introduce the "MAT-normalized throughput", $\gamma_{\text{MAT}}(\mathcal{P})$, by setting $\beta = s_{\max}$. Here, $0 \leq \gamma_{\text{MAT}}(\mathcal{P}) \leq 1$, and $s_{\max}$ is defined as the maximum number of successful simultaneous transmissions that can be supported by a given multi-hop network topology, for which all simultaneously transmitted data packets do not collide with each other. In other words, we normalize the computed aggregate throughput by maximum achievable throughput, which is the maximum aggregate data bit-rate characterized by the product of $s_{\max}$ and link rate R. Note that $s_{\max} \leq n/2$, since there could be at most $n/2$ number of simultaneous transmissions at any given time, due to the transceiver's half-duplex property. To further tighten the bound, we seek an *exact* maximum throughput by finding the optimal $s_{\max}$, via an optimization approach in Section 7.2.2. Compared to existing metrics, $\gamma_{\text{MAT}}(\mathcal{P})$ offers a clear quantitative indication of how close is the protocol's performance to what is best achievable, and is more useful in designing a better MAC protocol. Note, however, that we do not account for the protocol's fairness when computing $s_{\max}$. Nonetheless, it can be adopted for evaluating all classes of MAC protocols (e.g., contention-based MAC, schedule-based MAC, etc.), that do not enforce fairness.

7.2.2 The Binary Integer Linear Programming Formulation

Using a BILP optimization approach, we now explain how to compute $s_{\max}$ for a given network topology.

7.2.2.1 General Assumptions

We consider a static multi-hop wireless network with n homogenous nodes, and negligible propagation delay. Each node has a single omni-directional, half-duplex

transceiver with link rate R. All nodes are arbitrarily placed and they communicate using a single-channel. To obtain the maximum throughput, each node is assumed to be always backlogged and has packets destined to any of its one-hop neighbors. They also have a common and fixed communication range, which is the same as the interference range. Note that the formulation can also be modified accordingly if the interference range is assumed to be longer than the communication range. The channel is assumed to be error-free, and packet reception fails if and only if packets collide with each other. Although we ignore the effects of imperfect channel in this study, it can be considered by multiplying (7.4) with a factor of $(1 - p_{\mathrm{e}})$, where p_{e} is the packet error rate.

7.2.2.2 Problem Formulation

We denote the set of n nodes as $\mathcal{T} = \{1, \cdots, n\}$. A set of binary decision variables a_{ij} are introduced, in which $a_{ij} = 1$ if node i is scheduled to transmit to node j, and $a_{ij} = 0$ if otherwise. We define $\mathcal{N}(x)$ as the set of one-hop neighboring nodes of node x. We also define $a_x = \sum_{y \in \mathcal{N}(x)} a_{xy}$. Finally, we define M as a large number that is greater than $\sum_{k \in \mathcal{T}} a_k$. Our BILP formulation is presented as:

$$\text{maximize} \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{N}(i)} a_{ij} \tag{7.5}$$

subject to:

$$\sum_{j \in \mathcal{N}(i)} (a_{ij} + a_{ji}) \leq 1, \quad \forall i \in \mathcal{T} \tag{7.6}$$

$$a_{ij} + \sum_{k \in \mathcal{N}(j) \cup \{j\} \setminus \{i\}} (a_k) - (x_{ij} \cdot M) \leq 1, \quad \forall i \in \mathcal{T}, \forall j \in \mathcal{N}(i) \tag{7.7}$$

$$a_{ij} + x_{ij} = 1, \quad \forall i \in \mathcal{T}, \forall j \in \mathcal{N}(i) \tag{7.8}$$

$$a_{ij} = \{0, 1\};\ x_{ij} = \{0, 1\}, \quad \forall i \in \mathcal{T}, \forall j \in \mathcal{N}(i) \tag{7.9}$$

For the objective function in (7.5), we seek to maximize the total number of links (e.g., a_{ij}) that can be activated simultaneously. Since a node operates in half-duplex, constraint (7.6) ensures that it cannot transmit and receive at the same time. In addition, constraint (7.7) states that packet transmission from a sender node $i \in \mathcal{T}$ to its intended receiver node $j \in \mathcal{N}(i)$ is allowed if and only if the packet reception is free from interference at the receiver j (i.e., the receiver j and all its one-hop neighbors in $\mathcal{N}(j)$ must be inactive). Note that in (7.7), we also

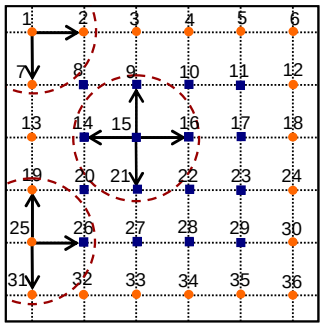


Figure 7.1: A square grid (6×6) topology used in our evaluations. Every node is placed at the grid intersection point. Each of the non-boundary nodes (square node) has four one-hop neighbors. In contrast, each of the boundary nodes (round node) has fewer than four neighbors; for example, the neighbors of node 1 are nodes 2 and 7.

introduce a binary variable x_{ij} (as defined in (7.8)) to ensure that no unnecessary constraint is imposed on the sum of a_k by (7.7) if a_{ij} is inactive[1]. The above BILP can be solved using a standard optimization solver such as CPLEX [82]. Finally, $s_{\max} = \sum_{x \in \mathcal{T}} a_x$ from any optimal solution found.

We now explain how the above formulation can be extended for the scenario when interference range is larger than the communication range. We define $\mathcal{N}_{\text{if}}(x)$ as the set of nodes that are located outside of node x's communication range, but can receive interference signal from node x's packet transmission. The following constraint can be included:

$$a_{ij} + \sum_{f \in \mathcal{N}_{\text{if}}(i),\ e \in \mathcal{N}(f)} (a_{ef}) \quad - (x_{ij} \cdot M) \leq 1, \quad \forall i \in \mathcal{T}, \forall j \in \mathcal{N}(i). \tag{7.10}$$

This means that when packet is transmitted from a sender $i \in \mathcal{T}$ to its intended receiver $j \in \mathcal{N}(i)$, the set of nodes $f \in \mathcal{N}_{\text{if}}(i)$ cannot successfully received any packets from their neighbors $e \in \mathcal{N}(f)$, due to interference effects.

7.3 Illustration Using Regular Structured Networks

We illustrate the use of our metric using both string and square grid networks (see Fig. 7.1), which are commonly used for evaluating MAC protocols. Unless stated otherwise, we adopt the same assumptions as in Section 7.2.2.1, for both topologies. Note that both topologies are *non-wraparound*.

[1]By having the term $x_{ij} \cdot M$, we ensure that the inequality (7.7) is still valid when $a_{ij} = 0$.

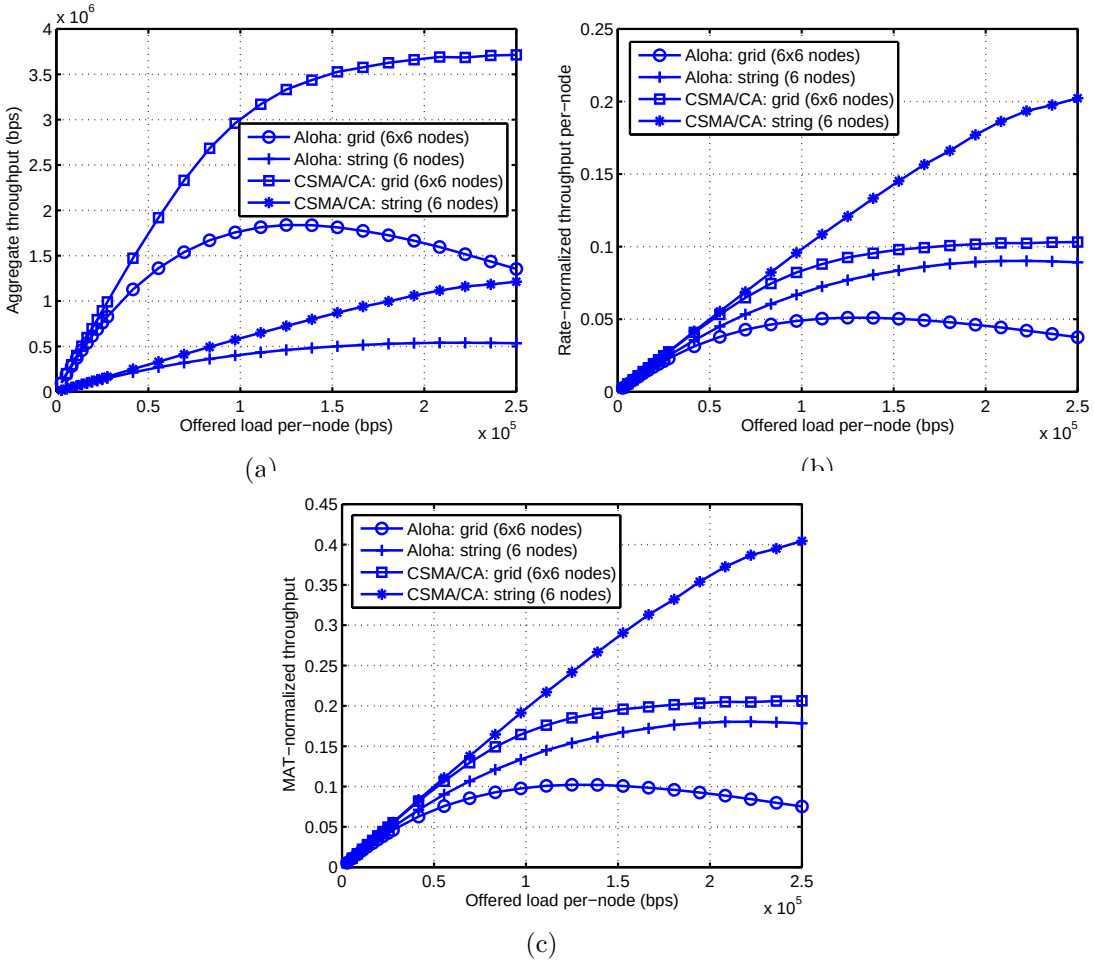


Figure 7.2: Throughput comparisons of Aloha and CSMA/CA MAC protocols in both string (6 nodes) and square grid (6×6 nodes) topologies, by using different throughput metrics: (a) aggregate throughput, (b) rate-normalized throughput per-node, and (c) MAT-normalized throughput.

7.3.1 Illustrating MAT-normalized throughput

Fig. 7.2(a)–7.2(c) show the use of different throughput metrics to evaluate Aloha and CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) protocols, in both string (6 nodes) and square grid (6×6 nodes) networks. Note that from CPLEX solutions, $s_{\max}$ for these string and grid topologies are 3 and 18, respectively (denoted by $s_{\max}^{\text{string}}(6)$ and $s_{\max}^{\text{grid}}(36)$ in Table 7.1 and Table 7.2, respectively). We also set $R = 1$ Mbps and $L_{\text{DATA}} = 1$ KB.

In Fig. 7.2(a), we observe that the aggregate throughput grows as the network size increases from 6 nodes to 36 nodes for both protocols, since a larger network could accommodate more simultaneous transmissions. However, the aggregate throughput metric does not illustrate the fact that a given MAC protocol experiences less contention in the string topology (i.e., due to less number of

Table 7.1: CPLEX's Solutions of $s_{\max}$ for String Topology

n	1	2	3	4	5	6	7	8	9	10
$s_{\max}^{\text{string}}(n)$	-	1	1	2	2	3	3	4	4	5
n	11	12	13	14	15	16	17	18	19	20
$s_{\max}^{\text{string}}(n)$	5	6	6	7	7	8	8	9	9	10

Table 7.2: CPLEX's Solutions of $s_{\max}$ for Square Grid Topology

d	1	2	3	4	5	6	7	8
$n = d \times d$	1	4	9	16	25	36	49	64
$s_{\max}^{\text{grid}}(n)$	-	2	4	8	11	18	22	32
d	9	10	11	12	13	14	15	16
$n = d \times d$	81	100	121	144	169	196	225	256
$s_{\max}^{\text{grid}}(n)$	37	50	56	72	79	98	106	128

neighbors), compared to its counterpart in grid topology. This, however, is shown in both Fig. 7.2(b)–7.2(c), in which it is realized that a MAC actually performs more efficiently in the string case, compared to the grid topology.

More importantly, the use of our MAT-normalized throughput in Fig. 7.2(c) offers even more insights, as it can show how close is a protocol's performance from the maximum possible throughput for a given network topology. As an example, it can now be appreciated from Fig. 7.2(c) that Aloha's peak throughput for a multi-hop string topology is actually quite close to its theoretical peak for single-hop networks, which is around 18% of the maximum achievable throughput. In contrast, the metric in Fig. 7.2(b) may give a misleading perception that the Aloha's peak throughput is much lower (around 0.09). Note that we have also simulated other string topologies with different sizes, and obtained similar conclusions.

7.3.2 s_{max} for both string and square grid topologies

Here, we derive the closed-form expressions of $s_{\max}$ for both string and square grid topologies. This will be useful for future MAC protocol designers to evaluate their protocols' performance using MAT-normalized throughput based on these two topologies, without the need to solve the BILP.

Theorem 1. $s_{\max}$ for a non-wraparound string topology of $\{n | n \in \mathbb{Z}^+$ and $n \neq 1\}$ nodes, $s_{\max}^{\text{string}}(n)$ is $\lfloor n/2 \rfloor$.

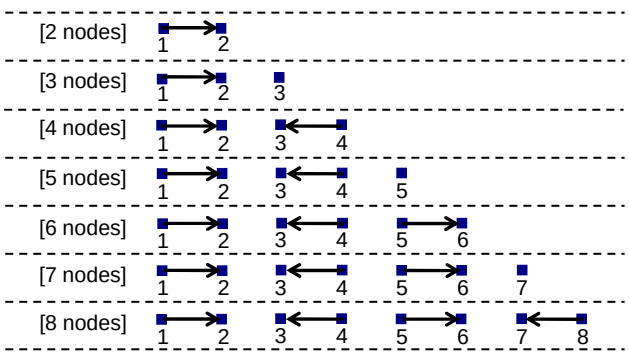


Figure 7.3: Several cases of string topologies, and their respective possible simultaneous transmission patterns that yield the optimal number of transmissions.

Proof: To ease our explanation, each node is given a unique ID (node IDs of 1 to n). We first show that the above expression is valid, when n is even. For $n = 2$, there is a single transmission. For $n = 4$, there can be 2 simultaneous transmissions; the transmission patterns are either $\{[1, 2], [4, 3]\}$ (as the 4-node case in Fig. 7.3), or $\{[2, 1], [3, 4]\}$, where $[x, y]$ denotes a sender x transmits to a receiver y. Note that the sender-receiver (S-R) node pair of 3-4 has an "inverse" transmission pattern, with respect to that of their adjacent S-R pair of 1-2. Similarly, for $n = \{6, 8, 10, \ldots\}$, every subsequent two nodes further along the topology forms an S-R pair that assumes an inverse transmission pattern from the preceding pair, so as to allow maximal successful simultaneous transmissions (see Fig. 7.3). From these repeating patterns, it can be seen that for even n, $s_{\max}^{\text{string}}(n) = n/2 = \lfloor n/2 \rfloor$. Finally, when n is odd, the continuous string of $n-1$ nodes can form $(n-1)/2$ disjoint S-R pairs, thus leaving a residual node at the end of the topology (see Fig. 7.3). To yield the optimal number of transmissions without causing any collision, those S-R node pairs can also assume the inverse transmission pattern. Hence, for odd n, $s_{\max}^{\text{string}}(n) = (n-1)/2 = \lfloor n/2 \rfloor$. ■

Theorem 2. $s_{\max}$ for a non-wraparound square grid topology of $\{n | n = d \times d, d \in \mathbb{Z}^+ \text{ and } d \neq 1\}$ nodes, $s_{\max}^{\text{grid}}(n)$ is:

$$s_{\max}^{\text{grid}}(n = d \times d) = \begin{cases} d^2/2, & \text{when } d \text{ is even,} \\ [d^2 - (d-2)]/2, & \text{when } d \text{ is odd.} \end{cases} \tag{7.11}$$

Proof: For a $d \times d$ square grid, we first show that $s_{\max}^{\text{grid}}(n = d \times d) = d^2/2$, when d is even. Now, let us assume that d number of nodes in the 1^{st}-row of the grid topology employ the inverse transmission patterns, as is done in the

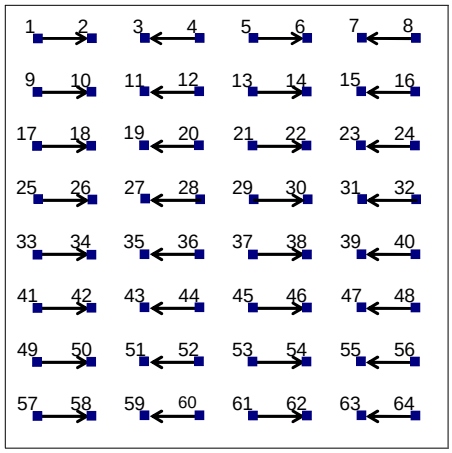


Figure 7.4: A possible simultaneous transmission pattern that yields the optimal number of transmissions of $s_{\max}^{\text{grid}}(64) = 32$ for the square grid topology when d is even (8×8 here).

string topology case, so as to maximize the number of successful simultaneous transmissions. In this case, there are $d/2$ S-R node pairs that can be allowed to transmit simultaneously. Then, in the 2^{nd}-row of the grid, the set of nodes that are directly under the same columns as the receiving nodes in the 1^{st}-row, cannot transmit because this would interfere with the packet receptions of those receiving nodes in the 1^{st}-row (e.g., nodes 10, 11, 14, and 15 cannot transmit in our example in Fig. 7.4). Instead, they could receive from their respective adjacent neighbors; thus, the d number of nodes in the 2^{nd}-row also have the same inverse transmission patterns, as those nodes in the 1^{st}-row. Following this argument, it can be seen that the remaining rows of the square grid, will also have the same transmission patterns as the 1^{st}-row. Hence, we have $s_{\max}^{\text{grid}}(n) = d/2 \times d$, when d is even.

Next, we show that $s_{\max}^{\text{grid}}(n = d \times d) = [d^2 - (d-2)]/2$, when d is odd. For a 3×3 grid, the optimal number of transmissions is 4; these transmissions are characterized by a "square" pattern, in which 4 non-conflicting S-R node pairs encompass an idle node, as shown in Fig. 7.5(a). For cases of $d = \{5, 7, 9, \ldots\}$, it is found that a generic transmission pattern, which consists of both square patterns (of size 3×3) and inverse transmission patterns, always yields an optimal number of transmissions. As grid sizes grow, it can accommodate multiple adjacent square patterns, which share common S-R node pairs at their squares' boundary, so as to maximize the number of simultaneous transmissions (see Fig. 7.5(b); note that those square patterns can start from any one of the grid's corners). As illustrated, these square patterns divide the grid into two regions (i.e., lower and upper

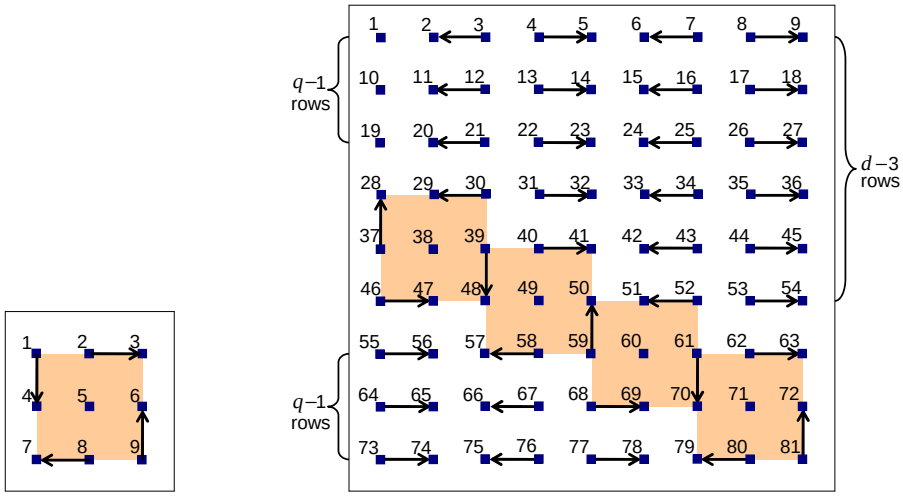


Figure 7.5: The possible simultaneous transmission patterns that yield the optimal number of transmissions in the square grid topology when d is odd: (a) 3×3 square grid topology, where $s_{\max}^{\text{grid}}(9) = 4$, (b) 9×9 square grid topology, where $s_{\max}^{\text{grid}}(81) = 37$.

regions), where those nodes can assume the inverse transmission patterns, so as to avoid interfering with the squares' transmissions. Let us denote q as the number of square patterns supported in a $d \times d$ square grid. For $d = \{3, 5, 7, 9, \ldots\}$, q would be $\{1, 2, 3, 4, \ldots\}$, respectively; thus, we can express $q = (d-1)/2$. Based on the above generic optimal transmission pattern for odd d, $s_{\max}^{\text{grid}}(n)$ can be computed as the summation of number of successful simultaneous transmissions in: (i) the square patterns (n_{sq}), (ii) the lower region (n_{lo}), and (iii) the upper region (n_{up}). Therefore, $s_{\max}^{\text{grid}}(n)$ is computed as,

$$
\begin{aligned}
s_{\max}^{\text{grid}}(n) &= n_{\text{sq}} + n_{\text{lo}} + n_{\text{up}} \\
&= (3q+1) + \sum_{i=1}^{q-1} i + \sum_{i=q-1}^{d-3} i \\
&= (3q+1) + \sum_{i=1}^{d-3} i + (q-1) \\
&= 4a + \frac{1}{2}(d-3)(d-2) \\
&= 4q + [(d-3)(d-2)]/2 = [d^2 - (d-2)]/2. \qquad \blacksquare
\end{aligned}
$$

We note that the transmission pattern in Fig. 7.5(b) is just one of the many possible optimal solutions. This means that multi-hop routing can be achieved by alternating between different optimal transmission patterns.

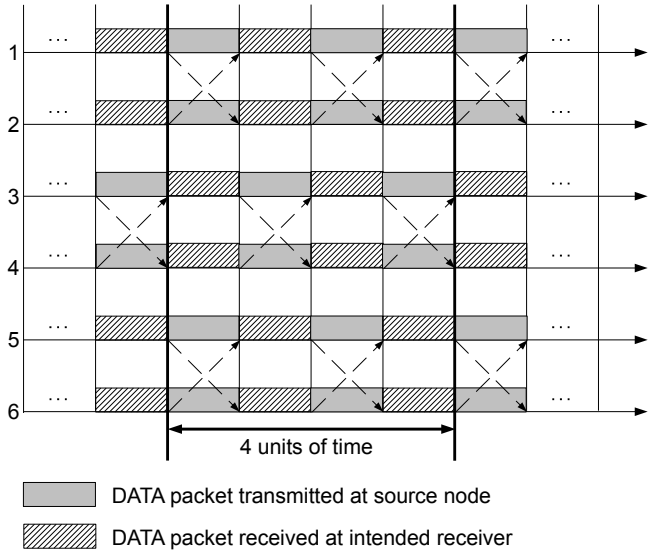


Figure 7.6: A possible transmission schedule that gives a maximum achievable throughput of $3R$, where R is the link rate, for a 6-node string topology in long propagation delay scenario. Here, there are 12 successful data packet receptions within 4 units of time.

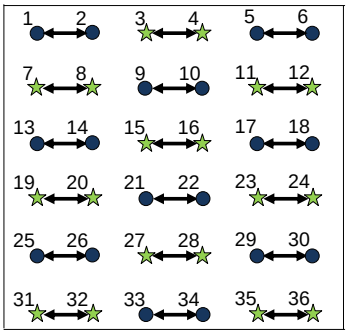


Figure 7.7: Illustration of achieving a maximum achievable throughput of $18R$ for a 6×6-node square grid topology in long propagation delay case. In each row of the grid, node pairs can assume the similar transmission schedules as done in Fig. 7.6. Note that the node pairs that are marked with stars have an orthogonal transmission schedule relative to their neighbors.

7.4 Evaluating BiC-MAC and ROPA protocols using MAT-Normalized Throughput Metric

We now extend the use of MAT-normalized throughput metric for long propagation delay networks. Specifically, we adopt the metric to evaluate our BiC-MAC [69] and ROPA [83] protocols using both string (6 nodes) and square grid (6×6 nodes) networks; here, both topologies follow the assumptions as described in Section 7.3, except that all inter-nodal delays are now no longer negligible. Recall that in the negligible delay scenario, the maximum achievable throughput for these string and square grid are $3R$ and $18R$, respectively, where R is the link rate. We next

show that these maximum achievable throughputs still remain the same when both topologies are used in a long propagation delay scenario. Specifically, to achieve an optimal channel utilization for the 6-node string network, any two adjacent nodes can simultaneously transmit to each other as shown in Fig. 7.6 (for simplicity, the inter-nodal delay is assumed to take 1 unit of time; also, the duration of a single data packet is equal to 1 unit of time). Note that the 3-4 node pair has an "orthogonal" transmission schedule relative to the schedules of neighboring 1-2 and 5-6 pairs (i.e., when both 1-2 and 5-6 node pairs start to transmit (receive), the 3-4 pairs will start to receive (transmit) at the same time), so as to ensure that its packet transmissions do not cause interference to the neighboring nodes' receptions. As can be seen, the maximum achievable throughput for the string network is $3R$, since there can be at most 12 successful packet receptions within 4 units of time. By extending the same argument into 2D grid network, we find a maximum achievable throughput of $18R$ for the 6×6-node square grid, as shown in Fig. 7.7. Here, the square grid has at most a total of 72 successful packet receptions within 4 units of time.

For both topologies in our simulation, the inter-nodal distance between any two adjacent communicating nodes is 2000 m. The acoustic propagation speed is 1500 m/s and the link rate R is 4800 bps. We also set $L_{\text{DATA}} = 6350$ bits. The rest of the protocol-specific settings such as control packet lengths, backoff window, etc., for BiC-MAC and ROPA will follow their respective settings in the simulation models described in Section 4.4.1 and Section 5.4.1. For benchmarking purpose, the MACA-U protocol is also simulated and Fig. 7.8 shows the MAT-normalized throughput comparisons. As shown, a given MAC protocol performs more efficiently in the string topology because there are less contention and interference from a smaller set of contending neighbors. We also find that ROPA outperforms both BiC-MAC and MACA-U protocols, which is consistent with our earlier finding in Section 5.4.2.1. Note that as the number of neighbors increases, ROPA has more opportunities for packet appending; as can be seen, the relative ROPA's throughput gain compared to BiC-MAC, has become more significant in the grid topology scenario. Compared to CSMA/CA (i.e., 3-way RTS/CTS/DATA handshake MAC) in Fig. 7.2(c), we can see that the similar 3-way handshake MACA-U's throughput is seriously degraded by the adverse effect of long propagation delay. Although ROPA and BiC-MAC perform much

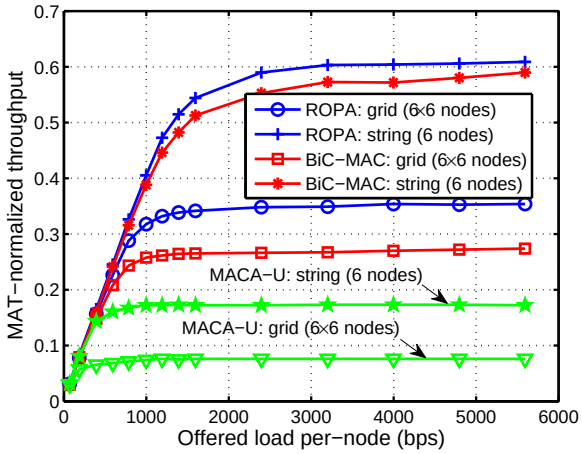


Figure 7.8: The MAT-normalized throughput comparisons of BiC-MAC, ROPA, and MACA-U MAC protocols in both string (6 nodes) and square grid (6×6 nodes) topologies, under a long propagation delay setting.

better than MACA-U, Fig. 7.8 also reveals that it is difficult for a practical MAC, especially a fully distributed contention-based protocol, to have a throughput performance close to the maximum achievable throughput. This is due to limiting factors such as backoff durations, control packet exchanges, packet collisions, etc.

7.5 Summary

In this chapter, we revisit the use of throughput metrics in studying MAC protocols in static multi-hop wireless networks. Three commonly used multi-hop throughput metrics: aggregate throughput, throughput per-node, and rate-normalized throughput per-node, are reviewed and their limitations are presented. First, these throughput metrics are not a generalization of the single-hop throughput metric, in which they do not seek to benchmark against the maximum achievable throughput. Hence, they do not provide as much intuition as the single-hop throughput metric, with regard to the performance relative to best achievable bit-rate. Second, many previous works only compare a proposed MAC protocol's throughput performance against a de facto MAC, which give a relative indication of performance improvement/degradation. Instead, a better approach would be benchmarking against the best achievable, which could offer an absolute performance measure.

To complement existing single-hop and multi-hop throughput notions, we first propose a unified normalized throughput expression. Since current multi-hop

metrics do not give much intuition on how close a MAC protocol's throughput performance is to the best achievable for a given network topology, we also introduce a new variant that benchmarks against the Maximum Achievable Throughput (MAT). The MAT-normalized throughput is characterized by the product of the number of maximum successful simultaneous transmissions under saturated traffic conditions and link rate. In order to find the maximum simultaneous transmissions in a given network topology, a BILP problem is formulated, which can then be solved via standard optimization solvers. Our MAT-normalized throughput metric not only allows for better performance comparison across different MAC protocols, but also serves as a useful guideline for system designers to decide whether any potential protocol enhancement is worth the effort.

We use the existing throughput metrics, as well as MAT-normalized throughput to evaluate Aloha and CSMA/CA MAC protocols in both string and square grid networks with negligible propagation delay. Unlike our proposed metric, the aggregate throughput metric fails to illustrate the fact that a given MAC protocol actually performs more efficiently in the string topology (due to less contention), compared to the grid case. More importantly, MAT-normalized throughput offers more insights as it compares a protocol's throughput to the best achievable. It can now be appreciated that Aloha's peak throughput for multi-hop string topology is actually quite close to its theoretical peak for single-hop networks, while the rate-normalized throughput per-node may give a misleading perception that the Aloha's peak throughput is much lower. Since solving the BILP could be computationally expensive for large networks, we also derive exact mathematical expressions for the maximum simultaneous transmissions for these two topologies.

Finally, we extend our metric to evaluate BiC-MAC, ROPA and MACA-U protocols in both string and grid topologies, under the presence of long propagation delay. For these regular structured topologies, we have shown that their maximum achievable throughputs still remain the same as the negligible delay case. We also point out that the maximum achievable throughput for other non-regular structured topologies in long propagation delay might not be the same as that of the negligible delay counterpart [84]; hence, their best achievable bit-rate could be difficult to compute and need to be considered in greater details.

Chapter 8

Conclusion and Directions for Future Research

This chapter summarizes and concludes our main contributions of this dissertation. We also outline several potential directions for our future research.

8.1 Research Contributions

This dissertation intends to address the issues of: (i) how to adapt the original MACA MAC protocol for use in multi-hop UWA networks, (ii) how to improve channel utilization of underwater handshaking-based MAC protocols, which in turn will offer both throughput and delay gains, (iii) how to accurately analyze the normalized saturation throughput of slotted BiC-MAC in single-hop networks, and (iv) how to better evaluate throughput performance of MAC protocols in static multi-hop wireless networks. The key contributions of this dissertation are:

1. In Chapter 3, we propose the MACA-U protocol (i.e., 3-way RTS/CTS/DATA handshaking-based MAC protocol with BEB backoff strategy), which is an adaptation of original MACA MAC protocol for use in multi-hop underwater acoustic networks. Three areas of improvement are identified and adapted for the long propagation delay, namely, the state transition rules, the packet forwarding strategy, and the backoff algorithm. Via simulation results, we have shown that the adapted MACA-U outperforms original MACA that applied blindly into underwater networks, in terms of throughput efficiency across all simulated offered load ranges. Unlike pure Aloha protocol, MACA-U is able to maintain a stable throughput as offered load further

increases. As opposed to the terrestrial MACAW's finding, we found that the MILD backoff algorithm actually performs worse than the BEB strategy, under the presence of long propagation delay. The simplicity and throughput stability of the adapted MACA-U make it a more appropriate reference MAC protocol, with which a more sophisticated underwater handshaking-based MAC protocol may benchmark its performance.

2. In Chapter 4, by exploiting the slow propagation speed of sound in water, we present a novel approach based on bidirectional, concurrent data packet exchange to improve data transmission efficiency; this would better utilize the channel than conventional MAC protocols, that often adopt a unidirectional data exchange method. To further amortize the high latency overhead, a packet bursting idea is also presented, where an S-R pair can exchange multiple rounds of bidirectional packet transmissions. Based on these strategies, we propose BiC-MAC, which is a single-channel, asynchronous handshaking-based MAC that achieves high channel utilization. Although the handshake is sender-triggered, an intended receiver can use that opportunity to initiate multiple rounds of bidirectional-concurrent transmissions if it has data packets in return. If it does not have any packets in return, a normal unidirectional transmission can still be performed. BiC-MAC is also designed with a versatile framework to support all possible modes of bidirectional transmissions. Unlike unidirectional MAC, both S-R nodes in BiC-MAC share only one set of communication overhead. Via single-hop and multi-hop simulations, we have shown that BiC-MAC significantly outperforms the normal unidirectional-based MAC protocols such as MACA-U, MACA-UPT, RIPT, etc., in both throughput and delay, while still offering a stable saturation throughput. The BiC-MAC variant that relaxes the assumption of having first-hop neighbors' knowledge, only slightly degrades the performance from the optimized BiC-MAC, and can still give remarkable performance gains compared to the existing unidirectional-based protocols.

3. In Chapter 5, we propose ROPA, which is a single-channel, sender-initiated handshaking-based MAC that does not require clock synchronization. It is based on a novel approach of reverse opportunistic packet appending, which exploits the opportunity of concurrent transmissions from multiple nodes in long propagation delay, so as to improve channel utilization. In each

handshake, an initiating sender can coordinate multiple first-hop neighbors (appenders) to transmit their appended data packets, with partial overlap in time. After the sender finishes transmitting its data packets to its own receiver (primary transmissions in forward path), it starts to receive the incoming appended data packets from different appenders (secondary transmissions in reverse path), which arrive in a collision-free packet train manner. Our packet exchange is more efficient than the conventional approach, that requires each of those neighbors to initiate a separate handshake that incurs its own overheads. ROPA is equipped with a versatile MAC framework that supports three possible data transmission modes. Both sender-initiated MACA with packet train and the receiver-initiated RIPT, can be considered as special cases of ROPA. From our single-hop and multi-hop simulations, we have shown that ROPA offers a stable saturation throughput, and provides significant gains in both throughput and delay compared to conventional handshaking-based MAC protocols such as MACA-U, MACA-UPT, slotted FAMA (only primary transmissions), as well as RIPT (only secondary transmissions). ROPA also surpasses BiC-MAC in terms of throughput and delay, at the cost of protocol complexity. Unlike BiC-MAC, it does not have stringent constraints on the packet size and inter-nodal separation distance.

4. In Chapter 6, we propose a novel analytical framework based on absorbing Markov chain to analyze the normalized saturation throughput of slotted BiC-MAC in single-hop networks, under error-free and error-prone channel conditions. Based on the key insight that time slotting will lose its effects when inter-nodal delay is much longer than a single control or data packet's duration, the analyzed slotted results can serve as an approximation for the unslotted counterpart. We model the protocol behavior of a single tagged node, as it attempts to exchange its backlogged batch of data packets, by using a bidirectional-concurrent transmission approach. From the resultant transition probability matrix and the fraction of time spent in each Markov state, we can compute the average batch service time (used to obtain the saturation throughput). Our model is the first to analyze bidirectional, concurrent packet exchange in high latency networks. From our comparisons with the simulated slotted BiC-MAC (all inter-nodal delays are fixed at a maximum value, so as to match our model's assumption) in both small and

large networks, we have shown that our model can give extremely accurate normalized saturation throughput results. Moreover, we also compare against the simulation results of both unslotted and slotted BiC-MAC, with actual inter-nodal delays. We show that our analytical model can reasonably well approximate the throughput performance of slotted BiC-MAC. This, in turn, can also be used to approximate the unslotted BiC-MAC's performance, since their throughput gap is quite close, especially when the number of nodes is small. We observed that this throughput gap will be widened when the number of nodes is large, which leads to less accurate estimation. Finally, we also propose another approximation approach that uses the information of actual inter-nodal delays, in the analytical expression. We show that it closely approximates the throughput of unslotted BiC-MAC with actual inter-nodal delays, for both network sizes.

5. In Chapter 7, we found that the three commonly used multi-hop throughput metrics: aggregate throughput, throughput per-node, and rate-normalized throughput per-node, do not offer as much intuition as the single-hop throughput metric, with regard to the performance relative to best achievable bit-rate. To complement both existing single-hop and multi-hop throughput notions, we first present a unified normalized throughput expression. Next, we propose the MAT-normalized throughput metric, which benchmarks against the maximum achievable throughput in a given static multi-hop topology. It is characterized by the product of link transmission rate and maximum number of successful simultaneous transmissions, which can be found via solving the formulated BILP problem. From our evaluation of MAC protocols in string and square grid networks with negligible propagation delay, our proposed metric is shown to be more effective and can offer more insights, compared to the existing multi-hop throughput metrics. To facilitate evaluation (without solving the BILP), we also derive exact mathematical expressions for the maximum simultaneous transmissions for these two topologies. Finally, we extend the MAT-normalized throughput metric to evaluate BiC-MAC, ROPA and MACA-U protocols in the string and grid topologies, under the presence of long propagation delay. For these regular structured topologies, we have shown that their maximum achievable throughputs still remain the same as the negligible delay counterpart.

8.2 Directions for Future Research

During the course of research, we have identified three key research areas that could enhance our MAC protocols: (i) energy-efficiency protocol design, (ii) dealing with node mobility, and (iii) integration of routing and MAC protocols. Each of these topics needs to be studied in greater detail.

8.2.1 Energy-efficiency of MAC Protocols

To prolong the network operation lifetime, minimizing energy consumption is an important issue, as communication nodes are typically powered by limited capacity battery. This becomes even more profound in underwater acoustic sensor networks, because it would be harder and more costly to replace battery for underwater nodes, compared to terrestrial sensor networks. We shall focus on energy conservation via MAC protocol design. While the works in [26, 32] deal with this issue by using a periodic sleep-listen schedules, the proposed MAC in [63] relies on a very-low-power wake-up tone receiver. However, their data transmission phase is generally inefficient. For future work, it would be interesting to study how BiC-MAC and ROPA can provide energy conservation, while not sacrificing too much of throughput performance. Currently, both of our handshaking protocols constantly listen to the channel for incoming control packets, so as to avoid packet collision, as well as schedule bidirectional transmission or packet appending. Clearly, this would result in more energy wasted on idle listening. Furthermore, the energy consumption of acoustic transducer in the idle, transmit and receive states, is different than that of the terrestrial modem counterpart [85]. It would be interesting to investigate how BiC-MAC and ROPA can adapt their transmission strategies, based on this energy consumption profile. We could also consider the incorporation of power control [22] technique into our protocols.

8.2.2 Handling of Node Mobility in MAC Protocols

Both BiC-MAC and ROPA are designed for static underwater acoustic networks; the "static" communication nodes are typically anchored at the seabed, and still subjected to limited sway distance, caused by underwater current. As explained, small guard times can be used to cater for this limited node movement. However, it would be useful to study how the protocols can be extended for handling a more

dynamic mobility scenario, such as the incorporation of autonomous underwater vehicle (AUV). For instance, a swarm of AUVs could communicate with each other during a military mission, or several AUVs perform periodical data sampling from static sensor nodes. The existing AUVs typically travel at the rate of up to 2.5 m/s [7]. Both of our protocols rely on the knowledge of inter-nodal delays (which are estimated either during network initialization or via each handshake by computing the round-trip time of control packet exchanges), to operate correctly. Thus, it will be interesting to examine the impact of mobility on the protocols' packet transmission behavior. For example, in BiC-MAC, the number of data packets exchanged between a moving S-R pair might no longer be the same as time progresses. So, the number of packets sent in each bidirectional round and its transmission timing need to be adjusted periodically according to latest inter-nodal distances. The AUV mobility pattern can also be utilized to better understand link breakage probability, which could offer an opportunity to further optimize the transmission strategies. In addition, the effects of node mobility model [86], which characterizes the sensor node movements due to oceanic currents, should be taken into consideration when designing networking protocols.

8.2.3 Integration of Routing and MAC Protocols

The functionality of routing protocols is to find a path from a source node to a destination (sink) node for packet forwarding. The criteria of selecting a forwarding node, highly depends on the application; for example, a common consideration would be minimizing the hop-count, so as to consume less network resources. For terrestrial sensor networks, there are some research efforts [87–89] that study how routing and MAC protocols can work closely together, by sharing certain information via cross-layer design technique; generally, this allows a more efficient packet forwarding. For underwater sensor networks, we are also interested to investigate how our MAC protocols can be enhanced and integrated with routing functionality, so as to offer lower end-to-end latency. The routing design should take advantage of the unique packet communication pattern offered by our protocols, i.e., bidirectional-concurrent transmission and reverse opportunistic packet appending. For instance, the bidirectional packet exchange allows a sensor node to transmit packets to its downstream node in a path, as well as receiving packets from the latter, which could be information broadcasted by the sink node.

Appendix A

Expression of $n_{1,j}$ for (6.15)

To ease the verification of our analytical model, we give the closed-form expression of $n_{1,j}$, $j = \{1, 2, \ldots, 11\}$. We have solved the matrix in (6.13) using MATLAB. For the convenience of notation, we also define $A = 2NW - 2W - 2N + 3$; $n_{1,j}$ are given as follows:

$$n_{1,1} = \frac{1}{abc} = \frac{2(N-1)(W-1)^2}{\delta_{\mathrm{C}}^3 \cdot (\frac{W-1}{W})^N \cdot A}. \tag{A.1}$$

$$n_{1,2} = \frac{1}{bc} = \frac{2(N-1)(W-1)^2}{W \cdot \delta_{\mathrm{C}}^3 \cdot (\frac{W-1}{W})^N \cdot A}. \tag{A.2}$$

$$n_{1,3} = \frac{1}{c} = \frac{1}{\delta_{\mathrm{C}}}. \tag{A.3}$$

$$n_{1,4} = \frac{d}{abc} = \frac{2(W-1)(N-1)(N-2)}{\delta_{\mathrm{C}}^2 \cdot A}. \tag{A.4}$$

$$n_{1,5} = \frac{de}{abc} = \frac{2(W-1)(-\delta_{\mathrm{C}}^2 + \delta_{\mathrm{C}} + 1)(N^2 - 3N + 2)}{A}. \tag{A.5}$$

$$n_{1,6} = \frac{f}{abc} = \frac{2(W-1)(1-\delta_{\mathrm{C}})(N^2-3N+2)}{\delta_{\mathrm{C}} \cdot A}. \tag{A.6}$$

$$n_{1,7} = \frac{g}{abc} = \frac{2(W-1)(\delta_{\mathrm{C}}-1)^2(N-1)(N-2)}{A}. \tag{A.7}$$

$$n_{1,8} = \frac{h}{abc} = \frac{N-2}{\delta_{\mathrm{C}} \cdot A}. \tag{A.8}$$

$$n_{1,9} = \frac{i}{abc} = \frac{(N-2)(1-\delta_{\mathrm{C}})}{A}. \tag{A.9}$$

$$n_{1,10} = \frac{j}{abc} = \frac{1}{\delta_{\mathrm{C}}^2}. \tag{A.10}$$

$$n_{1,11} = \frac{jk}{abc} = 1. \tag{A.11}$$

Bibliography

[1] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: Research challenges," *Ad Hoc Networks*, vol. 3, no. 3, pp. 257–279, May 2005.

[2] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li, "Research challenges and applications for underwater sensor networking," in *Proc. IEEE WCNC*, Las Vegas, Nevada, Apr. 2006, pp. 228–235.

[3] D. Bertsekas and R. Gallager, *Data Networks (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.

[4] F. Schill, U. R. Zimmer, and J. Trumpf, "Visible spectrum optical communication and distance sensing for underwater applications," in *Proc. Australasian Conf. Robotics and Automation*, Dec. 2004.

[5] N. Fair, A. D. Chave, L. Freitag, J. Preisig, S. N. White, D. Yoerger, and F. Sonnichsen, "Optical modem technology for seafloor observatories," in *Proc. IEEE OCEANS 2006*, Sep. 2006.

[6] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data collection, storage, and retrieval with an underwater sensor network," in *Proc. SenSys 2005*, Nov. 2005, pp. 154–165.

[7] J. Partan, J. Kurose, and B. N. Levine, "A survey of practical issues in underwater networks," in *Proc. ACM WUWNet*, Sep. 2006, pp. 17–24.

[8] D. B. Kilfoyle and A. B. Baggeroer, "The state of the art in underwater acoustic telemetry," *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 4–27, Jan. 2000.

[9] J. Rice, B. Creber, C. Fletcher, P. Baxley, K. Rogers, K. McDonald, D. Rees, M. Wolf, S. Merriam, R. Mehio, J. Proakis, K. Scussel, D. Porta, J. Baker,

J. Hardiman, and D. Green, "Evolution of Seaweb underwater acoustic networking," in *Proc. MTS/IEEE OCEANS*, Sep. 2000, pp. 2007–2017.

[10] E. M. Sozer, M. Stojanovic, and J. G. Proakis, "Underwater acoustic networks," *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 72–83, Jan. 2000.

[11] M. Chitre, S. Shahabudeen, and M. Stojanovic, "Underwater acoustic communications and networking: Recent advances and future challenges," *Marine Technology Society Journal*, vol. 42, no. 1, pp. 103–116, Spring 2008.

[12] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part I - carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Transactions on Communications*, vol. 23, no. 12, pp. 1400–1416, Dec. 1975.

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

[14] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LAN's," in *Proc. ACM SIGCOMM*, London, UK, Oct. 1994, pp. 212–225.

[15] C. L. Fullmer and J. J. Garcia-Luna-Aceves, "Floor acquisition multiple access (FAMA) for packet-radio networks," in *Proc. ACM SIGCOMM*, Cambridge, MA, USA, Oct. 1995, pp. 262–273.

[16] IEEE 802 LAN/MAN Standards Committee, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Standard 802.11, 1999 edition*, 1999.

[17] Z. J. Haas and J. Deng, "Dual busy tone multiple access (DBTMA) - A multiple access control scheme for ad hoc networks," *IEEE Transactions on Communications*, vol. 50, no. 6, pp. 975–985, June 2002.

[18] H. Doukkali and L. Nuaymi, "Analysis of MAC protocols for underwater acoustic data networks," in *Proc. IEEE Vehicular Technology Conference*, Stockholm, Sweden, May 2005, pp. 1307–1311.

[19] M. Molins and M. Stojanovic, "Slotted FAMA: A MAC protocol for underwater acoustic networks," in *Proc. IEEE OCEANS 2006*, May 2006.

[20] X. Guo, M. R. Frater, and M. J. 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, Apr. 2009.

[21] B. Peleato and M. Stojanovic, "Distance aware collision avoidance protocol for ad-hoc underwater acoustic sensor networks," *IEEE Communications Letters*, vol. 11, no. 12, pp. 1025–1027, Dec. 2007.

[22] J. M. Jornet and M. Stojanovic, "Distributed power control for underwater acoustic networks," in *Proc. IEEE OCEANS*, Sep. 2008.

[23] A. Kebkal, K. Kebkal, and M. Komar, "Data-link protocol for underwater acoustic networks," in *Proc. IEEE OCEANS 2005*, June 2005, pp. 1174–1180.

[24] S. Shahabudeen, M. Chitre, and M. Motani, "A multi-channel MAC protocol for AUV networks," in *Proc. IEEE OCEANS 2007-Europe*, June 2007.

[25] N. Chirdchoo, W. S. Soh, and K. C. Chua, "MACA-MN: A MACA-based MAC protocol for underwater acoustic networks with packet train for multiple neighbors," in *Proc. IEEE Vehicular Technology Conference*, Marina Bay, Singapore, May 2008, pp. 46–50.

[26] P. Xie and J.-H. Cui, "R-MAC: An energy-efficient MAC protocol for underwater sensor networks," in *Proc. International Conference on Wireless Algorithms, Systems and Applications*, Aug. 2007, pp. 187–198.

[27] N. Chirdchoo, W. S. Soh, and K. Chua, "RIPT: A receiver-initiated reservation-based protocol for underwater acoustic networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 9, pp. 1744–1753, Dec. 2008.

[28] S. Shahabudeen and M. Chitre, "Design of networking protocols for shallow water peer-to-peer acoustic networks," in *Proc. IEEE OCEANS 2005-Europe*, June 2005, pp. 628–633.

[29] M. Stojanovic, "Optimization of a data link protocol for an underwater acoustic channel," in *Proc. IEEE OCEANS 2005*, June 2005, pp. 68–73.

[30] P. Xie and J.-H. Cui, "Exploring random access and handshaking techniques in large-scale underwater wireless acoustic sensor networks," in *Proc. IEEE OCEANS 2006*, Sep. 2006.

[31] K. Y. Foo, P. R. Atkins, T. Collins, C. Morley, and J. Davies, "A routing and channel-access approach for an ad hoc underwater acoustic network," in *Proc. MTTS/IEEE TECHNO-OCEAN*, Nov. 2004, pp. 789–795.

[32] M. K. Park and V. Rodoplu, "UWAN-MAC: An energy-efficient MAC protocol for underwater acoustic wireless sensor networks," *IEEE Journal of Oceanic Engineering*, vol. 32, no. 3, pp. 710–720, July 2007.

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

[34] K. Kredo II, P. Djukic, and P. Mohapatra, "STUMP: Exploiting position diversity in the staggered TDMA underwater MAC protocol," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009, pp. 2961–2965.

[35] 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, Mar. 2000.

[36] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of reliable transport protocol over IEEE 802.11 wireless LAN: Analysis and enhancement," in *Proc. IEEE INFOCOM*, June 2002, pp. 599–607.

[37] J. W. Robinson and T. S. Randhawa, "Saturation throughput analysis of IEEE 802.11e enhanced distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 5, pp. 917–928, June 2004.

[38] X. J. Dong and P. Varaiya, "Saturation throughput analysis of IEEE 802.11 wireless LANs for a lossy channel," *IEEE Communications Letters*, vol. 9, no. 2, pp. 100–102, Feb. 2005.

[39] Q. Ni, T. Li, T. Turletti, and Y. Xiao, "Saturation throughput analysis of error-prone 802.11 wireless networks," *Wireless Communications and Mobile Computing*, vol. 5, no. 8, pp. 945–956, Dec. 2005.

[40] P. Chatzimisios, A. C. Boucouvalas, and V. Vitsas, "Performance analysis of IEEE 802.11 DCF in presence of transmission errors," in *Proc. IEEE ICC*, Paris, France, June 2004, pp. 3854–3858.

[41] D. Malone, K. Duffy, and D. Leith, "Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions," *IEEE/ACM Transactions on Networking*, vol. 15, no. 1, pp. 159–172, Feb. 2007.

[42] F. Daneshgaran, M. Laddomada, F. Mesiti, and M. Mondin, "Unsaturated throughput analysis of IEEE 802.11 in presence of non ideal transmission channel and capture effects," *IEEE Transactions on Wireless Communications*, vol. 7, no. 4, pp. 1276–1286, Apr. 2008.

[43] K. Kosek-Szott, M. Natkaniec, and A. R. Pach, "A simple but accurate throughput model for IEEE 802.11 EDCA in saturation and non-saturation conditions," *Computer Networks*, vol. 55, no. 3, pp. 622–635, Feb. 2011.

[44] L. F. Vieira, J. Kong, U. Lee, and M. Gerla, "Analysis of Aloha protocols for underwater acoustic sensor networks," in *Proc. ACM WUWNet*, Los Angeles, California, USA, Sep. 2006.

[45] Y. Xiao, Y. Zhang, J. Gibson, G. Xie, and H. Chen, "Performance analysis of Aloha and p-persistent Aloha formulti-hop underwater acoustic sensor networks," *Cluster Computing*, vol. 14, no. 1, pp. 65–80, 2011.

[46] A. A. Syed, W. Ye, B. Krishnamachari, and J. Heidemann, "Understanding spatio-temporal uncertainty in medium access with Aloha protocols," in *Proc. ACM WUWNet*, Montreal, Quebec, Canada, Sep. 2007, pp. 41–48.

[47] J. Ahn, A. Syed, B. Krishnamachari, and J. Heidemann, "Design and analysis of a propagation delay tolerant ALOHA protocol for underwater networks," *Ad Hoc Networks*, vol. 9, no. 5, pp. 752–766, July 2011.

[48] O. O. Aldawibi, B. S. Sharif, and C. C. Tsimenidis, "Distance awareness scheduling for single-hop underwater ad-hoc network," in *Proc. IEEE OCEANS 2009-Europe*, May 2009.

[49] S. Shahabudeen and M. Motani, "Performance analysis of a MACA based protocol for adhoc underwater networks: Short paper," in *Proc. ACM WUWNet*, Berkeley, California, Nov. 2009.

[50] Z. Zhou, Z. Peng, J.-H. Cui, and Z. Shi, "Analyzing multi-channel MAC protocols for underwater acoustic sensor networks," UCONN CSE Technical Report: UbiNet-TR08-02, Aug. 2008.

[51] A. A. Syed and J. Heidemann, "Contention analysis of MAC protocols that count," in *Proc. ACM WUWNet*, Woods Hole, Massachusetts, Sep. 2010.

[52] H. Matsuno, H. Ishinaka, and T. Shigeyasu, "Effect of propagation delay and RTS packet recognition time on MACA," *Electronics and Communications in Japan (Part I: Communications)*, vol. 88, no. 1, pp. 21–31, Jan. 2005.

[53] N.-O. Song, B.-J. Kwak, J. Song, and L. E. Miller, "Enhancement of IEEE 802.11 distributed coordination function with exponential increase exponential decrease backoff algorithm," in *Proc. IEEE VTC'03 Spring*, New Orleans, Louisiana, USA, Apr. 2003, pp. 2775–2778.

[54] S.-R. Ye and Y.-C. Tseng, "A multichain backoff mechanism for IEEE 802.11 WLANs," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 5, pp. 1613–1620, Sept. 2006.

[55] M. Erol, L. F. M. Vieira, and M. Gerla, "Localization with Dive'N'Rise (DNR) beacons for underwater acoustic sensor networks," in *Proc. ACM WUWNet*, Montreal, Quebec, Canada, Sep. 2007, pp. 97–100.

[56] X. Cao, J. Li, Y. Chen, and C. Qiao, "Assembling TCP/IP packets in optical burst switched networks," in *Proc. IEEE GLOBECOM*, Taipei, Taiwan, Nov. 2002, pp. 2808–2812.

[57] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks," in *Ad Hoc Networking*, C. E. Perkins, Ed. Boston, MA, USA: Addison-Wesley, 2001, pp. 139–172.

[58] C. E. Perkins, E. M. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF RFC 3561, July 2003.

[59] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference-aware fair rate control in wireless sensor networks," in *Proc. ACM SIGCOMM*, Pisa, Italy, Sep. 2006, pp. 63–74.

[60] K. Lu, J. Wang, D. Wu, and Y. Fang, "Performance of a burst-frame-based CSMA/CA protocol: Analysis and enhancement," *Wireless Networks*, vol. 15, no. 1, pp. 87–98, Jan. 2009.

[61] R. Riggio, D. Miorandi, F. De Pellegrini, F. Granelli, and I. Chlamtac, "A traffic aggregation and differentiation scheme for enhanced QoS in IEEE 802.11-based wireless mesh networks," *Computer Communications*, vol. 31, no. 7, pp. 1290–1300, May 2008.

[62] F. Cali, M. Conti, and E. Gregori, "IEEE 802.11 protocol: Design and performance evaluation of an adaptive backoff mechanism," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 9, pp. 1774–1786, Sep. 2000.

[63] A. Syed, W. Ye, and J. Heidemann, "Comparison and evaluation of the T-Lohi MAC for underwater acoustic sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 9, pp. 1731–1743, Dec. 2008.

[64] A. F. Harris III and M. Zorzi, "Modeling the underwater acoustic channel in ns2," in *Proc. of the 2nd International Conference on Performance Evaluation Methodologies and Tools 2007*, Oct. 2007, pp. 22–27.

[65] LinkQuest Inc., http://www.link-quest.com.

[66] H.-H. Ng, W.-S. Soh, and M. Motani, "MACA-U: A media access protocol for underwater acoustic networks," in *Proc. IEEE GLOBECOM*, New Orleans, Louisiana, Dec. 2008.

[67] C.-C. Hsu, K.-F. Lai, C.-F. Chou, and K. C.-J. Lin, "ST-MAC: Spatial-temporal MAC scheduling for underwater sensor networks," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009, pp. 1827–1835.

[68] T. Nandagopal, T.-E. Kim, X. Gao, and V. Bharghavan, "Achieving MAC layer fairness in wireless packet networks," in *Proc. ACM MobiCom 2000*, Boston, MA, USA, Aug. 2000.

[69] H.-H. Ng, W.-S. Soh, and M. Motani, "BiC-MAC: Bidirectional-concurrent MAC protocol with packet bursting for underwater acoustic networks," in *Proc. IEEE OCEANS 2010*, Sep. 2010.

[70] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "The challenges of building mobile underwater wireless networks for aquatic applications," *IEEE Network*, vol. 20, no. 3, pp. 12–18, May 2006.

[71] I. Akyildiz, Y. Altunbasak, F. Fekri, and R. Sivakumar, "Adaptnet: An adaptive protocol suite for the next-generation wireless internet," *IEEE Communications Magazine*, vol. 42, no. 3, pp. 128–136, Mar. 2004.

[72] C. M. Grinstead and J. L. Snell, *Introduction to Probability.* Providence, RI: American Mathematical Society, 1997.

[73] X. Yang and N. H. Vaidya, "Priority scheduling in wireless ad hoc networks," in *Proc. ACM MobiHoc*, Lausanne, Switzerland, June 2002, pp. 71–79.

[74] J. So and N. H. Vaidya, "Multi-channel MAC for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver," in *Proc. ACM MobiHoc*, Roppongi Hills, Tokyo, Japan, May 2004, pp. 222–233.

[75] R. Mangharam and R. Rajkumar, "MAX: A maximal transmission concurrency MAC for wireless networks with regular structure," in *Proc. BROADNETS 2006*, San José, California, USA, Oct. 2006.

[76] A. El-Hoiydi, "Aloha with preamble sampling for sporadic traffic in ad hoc wireless sensor networks," in *Proc. IEEE ICC*, New York, USA, Apr. 2002, pp. 3418–3423.

[77] A. Jayasuriya, S. Perreau, A. Dadej, and S. Gordon, "Hidden vs. exposed terminal problem in ad hoc networks," in *Proc. Australian Telecommunication Networks and Applications Conference*, Sydney, Australia, Dec. 2004.

[78] J. Li, Z. J. Haas, M. Sheng, and Y. Chen, "Performance evaluation of modified IEEE 802.11 MAC for multi-channel multi-hop ad hoc network," in *Proc. IEEE Advanced Information Networking and Applications (AINA)*, Mar. 2003, pp. 312–317.

[79] A. Baiocchi, A. Todini, and A. Valletta, "Why a multichannel protocol can boost IEEE 802.11 performance," in *Proc. ACM MSWiM*, Venice, Italy, Oct. 2004, pp. 143–148.

[80] M. Veyseh and J. J. Garcia-Luna-Aceves, "Parallel interaction medium access for wireless ad hoc networks," in *Proc. IEEE Computer Communications and Networks (ICCCN)*, St. Thomas, U.S. Virgin Islands, Aug. 2008.

[81] J. J. Garcia-Luna-Aceves and C. L. Fullmer, "Floor acquisition multiple access (FAMA) in single-channel wireless networks," *Mobile Networks and Applications*, vol. 4, no. 3, pp. 157–174, 1999.

[82] ILOG CPLEX 10.0 User's Manual, http://www.ilog.com/products/cplex.

[83] H.-H. Ng, W.-S. Soh, and M. Motani, "ROPA: A MAC protocol for underwater acoustic networks with reverse opportunistic packet appending," in *Proc. IEEE WCNC*, Sydney, Australia, Apr. 2010.

[84] M. Chitre, M. Motani, and S. Shahabudeen, "A scheduling algorithm for wireless networks with large propagation delays," in *Proc. IEEE OCEANS'10*, May 2010.

[85] A. F. Harris III, M. Stojanovic, and M. Zorzi, "When underwater acoustic nodes should sleep with one eye open: Idle-time power management in underwater sensor networks," in *Proc. ACM WUWNet*, Los Angeles, CA, USA, Sep. 2006, pp. 105–108.

[86] A. Caruso, F. Paparella, L. F. M. Vieira, M. Erol, and M. Gerla, "The meandering current mobility model and its impact on underwater mobile sensor networks," in *Proc. IEEE INFOCOM 2008*, Phoenix, Arizona, USA, Apr. 2008, pp. 221–225.

[87] M. Rossi and M. Zorzi, "Integrated cost-based MAC and routing techniques for hop count forwarding in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 4, pp. 434–448, Apr. 2007.

[88] D. Ferrara, L. Galluccio, A. Leonardi, G. Morabito, and S. Palazzo, "MACRO: An integrated MAC/routing protocol for geographic forwarding in wireless sensor networks," in *Proc. IEEE INFOCOM 2005*, Miami, Florida, USA, Mar. 2005, pp. 1770–1781.

[89] S. Du, A. K. Saha, and D. B. Johnson, "RMAC: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM*, Anchorage, Alaska, USA, May 2007, pp. 1478–1486.

List of Publications

- H.-H. Ng, W.-S. Soh, and M. Motani, "An Underwater Acoustic MAC Protocol Using Reverse Opportunistic Packet Appending", *Computer Networks*, under review.

- H.-H. Ng, W.-S. Soh, and M. Motani, "A Bidirectional-Concurrent MAC Protocol with Packet Bursting for Underwater Acoustic Networks", *IEEE Journal of Oceanic Engineering*, to appear.

- H.-H. Ng, W.-S. Soh, and M. Motani, "On the Throughput Comparisons of MAC Protocols in Multi-hop Wireless Networks", *IEEE Communications Letters*, vol. 15, no. 12, pp. 1398-1401, Dec. 2011.

- H.-H. Ng, W.-S. Soh, and M. Motani, "BiC-MAC: Bidirectional-Concurrent MAC Protocol with Packet Bursting for Underwater Acoustic Networks", in *Proc. MTS/IEEE OCEANS*, Seattle, Washington, United States, Sept. 2010.

- H.-H. Ng, W.-S. Soh, and M. Motani, "ROPA: A MAC Protocol for Underwater Acoustic Networks with Reverse Opportunistic Packet Appending", in *Proc. IEEE WCNC*, Sydney, Australia, Apr. 2010.

- H.-H. Ng, W.-S. Soh, and M. Motani, "MACA-U: A Media Access Protocol for Underwater Acoustic Networks", in *Proc. IEEE GLOBECOM*, New Orleans, Louisiana, USA, Dec. 2008.