UNIVERSITY OF CALIFORNIA
SANTA CRUZ

AN ANALYSIS OF CONTENTION RESOLUTION TECHNIQUES IN QSMA

A thesis submitted in partial satisfaction of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE AND ENGINEERING

by

Pranav Raghavan

March 2021

The Thesis of Pranav Raghavan
is approved:

Professor J. J. Garcia-Luna-Aceves, Chair

Professor Katia Obraczka

Professor Martine Schlag

Quentin Williams
Acting Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

An Analysis of Contention Resolution Techniques in QSMA

by

Pranav Raghavan

This thesis introduces QSMA-CR, a MAC layer protocol that leverages carrier sensing to achieve lower latency than its predecessors: QSMA [4] and ALOHA-QS [3]. QSMA-CR establishes a distributed queue to maintain fairness and achieve high throughput. Each node sends data frames within their reserved slot every queue cycle. After every cycle is a joining period in which unjoined nodes attempt to make a reservation. In QSMA and ALOHA-QS, nodes have latency due to a lack of a contention resolution algorithm that optimizes the join process. QSMA-CR has a global and node-level state, enabling smarter contention resolution that allows high throughput and lower latency. We used a 10-minute experiment with 50 nodes to show that QSMA-CR can achieve lower latency than QSMA achieving similar throughput.

Chapter 1

Introduction

Over the last few decades, many researchers have worked on medium-access control (MAC) protocols. The simplicity of ALOHA [9] created room for many MAC protocols, broadly categorized as contention-based and contention-free protocols. In contention-free protocols, nodes have to make reservations by sending signaling packets before sending any data. Signaling packets are exchanged during reservation periods and with some contention, but this keeps the data transmissions **contention-free**. We discuss some critical contention-free protocols that make QSMA-CR possible in chapter 2. Garcia-Luna-Aceves et al. introduced one such contention-free MAC protocol called QSMA [4] which establishes and maintains a distributed transmission queue among the nodes sharing the common channel while using carrier sensing to reduce collisions. Nodes that have reserved their slots will send packets in the order that they joined the queue. This thesis puts forward Queue-Sharing Multiple Access with Contention Resolution (QSMA-CR), decreasing the latency and maintaining the same bandwidth guarantees as QSMA. To evaluate QSMA-CR, we compare it to QSMA and ALOHA-QS in chapter 4. We decrease the delay of a join by modulating the probability of a successful join during the join turn. We modulate the probability of a successful

join by time slotting the reservation period and developing a contention resolution algorithm discussed in detail in the chapter 3. Although QSMA and QSMA-CR have made significant strides in incorporating carrier sensing into ALOHA-QS, there is still room for improvement, and we have detailed that in chapter 5.

In QSMA-CR, the queue sharing part is unchanged from QSMA because time-slotting is a great way to maintain fairness, goodput, and bandwidth guarantees in a wireless or wired network with a shared channel. In this protocol, nodes send their data transmissions in cycles, with every node in the transmission group getting their reserved turn. All nodes can quickly tell one turn from the next without clock synchronization because of carrier sensing. After every cycle of queue turns is the contention period or join turn. The join period allows a particular node outside the transmission group to send a control packet to join it. The join period has been the central focus of this thesis. All the nodes can set the ack bit in the header of their data transmission to acknowledge a successful join. However, this method is insufficient if multiple nodes need to join the transmission group. This thesis maintains the single join idea and improves it by modulating the probability of success and maintaining fairness by prioritizing nodes that wanted to join first.

The most basic forms of ALOHA, as well as ALOHA-QS, do not use carrier sensing. Nodes use exponential backoff to join the queue resulting in high latency in times of contention. ALOHA-QS does not allow for more than one node to be in contention for the join because all joining nodes, nodes with data transmit, send their signaling packets at the start of the joining slot. With the addition of carrier sensing in QSMA, more than one node can contend for the joining slot. All nodes in contention will have an equal probability of joining the queue irrespective of how early the node had data to send. ALOHA-QS, QSMA, and QSMA-CR all use exponential backoff upon failure. QSMA implemented a random backoff

local to the joining period, i.e., all nodes draw a number, H, from a uniform random distribution from zero and three inclusive and send their signaling packet at exactly $H\tau$ seconds after the start of the joining period. If two nodes pick the same random backoff, H, this results in a **collision**.

Carrier sensing is most helpful in the joining or contention period, and we expect to see significant returns from changing the join turn of QSMA. It has been used in many contention-based protocols before and is even a part of widely used MAC protocol standards (e.g., WiFi and WiMAX). We borrow a simple idea from 1-persistent CSMA to improving our contention period. Upon collision, nodes wait a random amount of time before polling the wire and transmitting again. The randomness is helpful because multiple nodes might want to use the wire after the ongoing transmission. The probability that a single station will pick the lowest backoff to successfully send a signaling packet is high no matter how many stations are in contention. Our goal with the contention period is to enable one successful control packet transmission out of many nodes in contention. In QSMA-CR, nodes wait a random period of time after the start of the joining period. We also have the luxury of using discrete intervals of the transmission delay based on our assumption that nodes are close to one another and using a shared medium. Random backoff, by definition, is random and would offer no priority to nodes that wanted to send packets earlier. As a part of our contention resolution efforts, we prioritized nodes that want to send packets first.

This thesis examines the benefits of leveraging carrier sensing even further to handle contention among nodes waiting to join the queue. In QSMA-CR, many nodes can be in contention, just like QSMA. The key difference is that we attempt a type of contention resolution where 1) nodes with data to send first have priority and 2) periods of higher contention and lower contention periods are treated dif-

ferently. We retain the idea of time slotting in the contention period from QSMA and modulate the number of slots using our contention resolution algorithm to treat periods of high and low contention differently. Increasing the number of mini-slots globally in all join turns has adverse effects on the throughput, so we use a simple algorithm to modulate it instead. When enough nodes are already in the queue, we do not have to draw from a uniform random distribution like QSMA. Instead, we can use the "turn number" when a node had a packet to send or exited exponential backoff to prioritize nodes that exited exponential backoff first.

Chapter 2

Related Work

One of the earliest MAC protocols, ALOHA [1], was developed in 1970 to provide data communication between the University of Hawaii campuses. ALOHA is the most straightforward medium access protocol. In the version described initially by Abramson, every device transmits its packets independent of any other device or any specific time. After a random timeout, the device sends the data packet; it then sets a timeout to receive an acknowledgment (ACK). If an ACK is not received, the device assumes a collision with a packet transmitted by some other device. This process continues until a successful transmission. Under a specific set of assumptions, Abramson showed that such a channel's effective capacity is $\frac{l}{2e}$. Although ALOHA-based schemes have the advantages of simplicity and low overheads, it suffers from poor throughput performance resulting from their blind transmission and purely random strategy. A variation of ALOHA, Slotted ALOHA [9] uses slots to improve ALOHA's efficiency. In Slotted ALOHA, physical layers are responsible for clock synchronization, requiring senders to transmit only at the slot intervals. Thus, the protocol vulnerability to multiple access interference (MAI) reduces from 2 packet lengths to 1. Hence packets overlap entirely or not at all. As a result, slotted ALOHA doubles the maximum throughput com-

pared to ALOHA, which retains vulnerability to two packets. Many improvements on ALOHA and Slotted ALOHA [10] assume time slotting and frame synchronization, which relies on physical layer assistance. These approaches suffer from two main limitations. One is the need for time synchronization and the use of fixed predefined frames. Clock synchronization is challenging in topologies like multihop networks, especially where propagation delay is long.

Xu and Campbell [13] introduced Distributed Queue Random Access Protocol (DQRAP), the first example of a protocol that has a shared transmission group. DQRAP assumes that the channel is time-slotted and that each time slot consists of data slots and multiple control mini-slots. Many such protocols have been created and studied since then. In these schemes, each node first contends in a contention period to send a reservation request and then transmit in the reserved time slot without contention. The main drawback of this approach is that the physical layer is responsible for time slotting the channel. QSMA-CR uses some of the ideas developed initially in DQRAP, but it does not rely on the time slotting of the channel. The transmission group, in this thesis, is a **Distributed Queue** - an ordered list of nodes waiting for access to the transmission channel.

Xie et al. defined another protocol that uses reservations [12]. In this protocol, the transmission channel consists of frames, and each one begins with a set of mini-slots and ends with a variable number of fixed-length data slots. Mini-slots control frames in a way that each node in the network is assigned a unique mini-slot, and if the node has data to transmit, it specifies this in the mini-slot [5, 6]. We used their terminology to describe a single division of the contention period. In QSMA, unjoined nodes can use any mini-slot. The contention resolution algorithm determines which slot to use and how many slots there are. The node with the earliest mini-slot joins the queue.

Carrier Sense Multiple Access with collision avoidance (CSMA-CA) [11] extended upon ALOHA and slotted ALOHA by endowing receivers with carrier sensing (ability to listen to the channel). They did this by polling the wire before sending a packet. CSMA did not need to clock synchronization at the channel because nodes know when another is sending packets. When a collision occurs in 1-persistent CSMA, nodes draw from a uniform random distribution to determine when the retry attempt should be. Unless two nodes have the same random back-off, there will be a successful transmission on the retry. It is also well understood that as the maximum number in the uniform distribution increases, the probability of success also increases at the cost of throughput. By adding this simple but powerful idea into the QSMA-CR join turn, we could increase the probability of successful join when contention is high. However, during periods of low contention, we could lower the probability of success.

Group Allocation Multiple Access (GAMA) [8] improves on DQRAP by eliminating the need for time slotting and the use of control mini-slots. QSMA-CR also uses dynamic reservations to improve efficiency and ensure that no collisions involving data packets occur. Each cycle has contention-free data frames as well as a contention-period where nodes join the transmission groups. Once a station is in the transmission group, it can transmit a packet during each cycle. CARMA-NTS [2] integrates collision avoidance and resolution in the contention period of GAMA, which results in each contention period having a successful join. QSMA-CR does not guarantee a successful join because this would increase every single join slot's length, costing clients bandwidth.

Sync-less Impromptu Time-Divided Acess (SITA) [7] combines the advantages of TDMA with the simplicity and robustness of CSMA/CA. Jakllari et al. proposed a "transmission group" to provide efficiency and stability under all condi-

tions while remaining simple to implement. The joining station must also reserve the bandwidth. A transmission group is a set of stations that have contention-free access to the transmission channel. In ALOHA-QS, QSMA, and QSMA-CR, this contention-free period is the queue itself. The control packet exchange that occurs during the joining slot facilitates the joining of nodes into the transmission group. The reservation is not for a specific bandwidth but to be included in the transmission group. As the number of elements in the transmission group increases, the bandwidth guarantee reduces. Only members of the transmission group are allowed to send data, and group members take turns transmitting.

One limitation of many prior MAC protocols based on distributed queues is that, in order to build and maintain such queues, they must rely on either: (a) time slotting and transmission frames that require clock synchronization, or (b) explicit signaling between specific transmitter-receiver pairs that requires senders to know whether intended receivers are present before the transmission queue can be built. To alleviate this problem, ALOHA-QS was created by JJ et al. [3] The basic idea is to time-slot the channel so that each node has its designated "turn" in the queue. After every "cycle" of the queue, nodes are allowed to join the queue. Additionally, nodes only join the queue when their request to join is in the last queue turn. Once nodes in the queue have successfully received a control packet, they will set their "ack" bit in their transmissions in the queue—the ack bit signals to the joining node that their control packet was successfully received. New nodes that join the queue take their place at the end of the queue. All the transmissions between nodes in the network are broadcasts. Broadcasts are required because all the nodes need to be notified of every join to know when their next queue turn will be. It is also required for the joining node to decypher their acknowledgments from the transmission group. QSMA is a derivative of this simple ALOHA variant

with carrier sensing.

QSMA's carrier sensing allows all the nodes to move forward to the next queue turn quickly [4] One of the most significant shortcomings of ALOHA-QS is that in any given join turn, only one node is allowed to be in contention to join. If more than one nodes need to use the join turn it will result in a collision because all the nodes send their signaling packet together. QSMA addresses this shortcoming by time slotting the join period into four transmission delay, τ s, sized mini-slots. QSMA's use of the joining period is similar to 1-persistent CSMA's handling of collisions. In CSMA, stations involved will retry their transmission after waiting a random period of time. In QSMA, that random time can be between 0 s and 3τ s in intervals of τ s. QSMA-CR analyzes the benefits of modulating the number of "mini-slots" depending on contention in the recent joining periods. Once the queue is large enough, we can prioritize nodes that wanted to send data first instead of relying on randomness. Much like CSMA, nodes will be able to carrier sense when another node is joining and avoid collisions. When nodes are assigned the same backoff, collisions take place. In all other cases: there will be a single node with the lowest random backoff that will successfully send its signaling packet to join the queue. This thesis extends upon QSMA by adding 1) the ability to adapt according to traffic conditions in the contention period and 2) prioritizing nodes that wanted to send data first.

Chapter 3

QSMA-CR

The design of the QSMA-CR protocol is similar to that of QSMA in the contention-free period. Each cycle consists of a sequence of one or more queue turns (i.e., uncontested transmissions by stations that already joined the queue) followed by a contention-filled joining period during which stations outside the transmission group attempt to send signaling packets to join it. In ALOHA-QS, QSMA and QSMA-CR stations back off exponentially on failure, i.e., another node has utilized the joining slot either with a lower or equal random backoff. The protocol works in cycles, continuing until there are one or more nodes in the queue. During the ALOHA-QS contention period, nodes send packets at will without using carrier sensing. As soon as the contention period starts, all nodes send their control packets together, failing if more than one node is in contention at a given join slot. With the addition of carrier sensing in QSMA, the contention period can accommodate more nodes into contention than ALOHA-QS. However, we believe there is room for improvement.

Mini-slots is the term that we have chosen to represent one transmission delay-sized interval of the join slot. The number of mini-slots available or the "maximum join slot", H_{MAX}, for random backoff can also be modulated depending on the

previous join turns. If there has been a failed join slot (collision), we increase the number of mini-slots available to joining nodes. Successful join means that a single node had the lowest backoff as a product of our distributed contention resolution algorithm. An empty join turn means that no node sent its signaling packet after waiting for $H_{MAX}\tau$ seconds. We developed a simple algorithm that modulates H_{MAX} during every cycle. If there is a successful or empty join turn, we can reduce the joining slot's length to facilitate more data transmission bandwidth. We lower the maximum mini-slots unless it is already at the lowest allowable maximum mini-slot, four. Alternatively, if a collision occurred in the join slot, we can increase the join slot's length to facilitate reducing the probability of failure on the next turn. Again we hard code the highest allowable maximum mini-slot, sixty-four, so that the join slot is never long enough to compromise bandwidth guarantees.

Our contention resolution algorithm relies on modulating the maximum number of mini-slots in a join period. In simple terms, **modulating maximum mini-slots** means changing the highest number, H_{MAX}, that can be drawn from a uniform random distribution while computing CSMA random timeout. Modulating H_{MAX} changes the probability of success when joining, making this analogous to a knob that the distributed algorithm can tweak. We make these modulations as a result of previous join periods. When H_{MAX} is changed, all nodes have to be informed regardless of membership in the transmission group. It is fair to assume that all nodes within the transmission group can observe every previous join period to calculate the maximum number of join slots. However - a node (outside the transmission group) can come online after the others. Not having received previous transmissions - it would be impossible for any node to figure out the maximum mini-slots for that join slot. Thus, a header field S holding the H_{MAX}

needs to be added to the QSMA header to convert it into a QSMA-CR header. Any station outside the transmission group will know the max random timeout to contest other nodes attempting to join.

In contrast with ALOHA-QS and QSMA, QSMA-CR uses modulation of H_{MAX} to ensure fewer collisions during the contention period while maintaining bandwidth guarantees. A node with the shortest backoff time will join the queue and send its data packet during the subsequent cycle's last turn. The joined station maintains ownership of this transmission period until it sends a termination bit in its header. All nodes that exited backoff during a given cycle are in contention during the joining period. The node with the lowest backoff time will join the transmission queue, and all other contenders must back off exponentially. Upon joining the queue, nodes can send data transmissions without collisions because their slot in the queue cycle is reserved, and transmission proceeds at a guaranteed bandwidth.

In QSMA-CR, use a metric called "backoff exit turn," representing the turn in which the joining node had data to send or exited exponential backoff (if it has already failed to join before). This metric helps us build priority into the join turn - i.e., if a node had data to send during the 3rd turn, it should naturally join the queue before the node that had data to send during the 7th turn. When the number of nodes in the transmission group is low enough, nodes default to drawing randomly from a uniform distribution like CSMA. For example, if there are only two nodes in the queue, the backoff exit turn can only be 1 or 2; however, they can get a more considerable variance in the random backoff if they generated a random number between 0 and 3 instead. This random strategy ensures some fairness because nodes will all have an equal probability of success. However, since nodes with earlier backoff exit turn have no priority, this technique is unfair

and random when the queue size is large enough. Such a protocol could *neglect* nodes for many cycles, which is unfair. To avoid neglecting nodes, we maintain a local state in the joining node to represent backoff exit turn instead of generating random numbers.

Every contention period in QSMA always works the same way. Nodes draw randomly from a uniform distribution between 0 and 3, H, and send their signaling packet at exactly $H\tau$ seconds after the start of the join period. In QSMA-CR maximum mini-slots, H_{MAX}, is variable, and so is the probability of a successful join. Backoff number H is an integer between 0 and H_{MAX} and dictates how much a node waits after the start of the join period to send a signaling packet. H_{MAX} starts at 16, and it gets incremented by four every time there is a failure or empty join slot, and it gets decremented by four every time there is a success. Intuitively, this means that the H_{MAX} is lower when there is not much contention and higher when there is a lot of it. The lowest H_{MAX} is four, and the highest H_{MAX} is 64, meaning that these values will not get incremented or decremented past this four to 64 threshold. The actual backoff itself, which is a multiple of τ, depends on when the node intended to join the queue. One interesting thing of note is that when contention is low, we default to QSMA's number of max join slots keeping out bandwidth guarantees on par with QSMA.

Each node independently computes its backoff number H based on local state variables or random number generators. However, nodes first need to know the H_{MAX}, which is the highest backoff there can be. In order to figure this out, nodes have to know the result of the previous join turn. In QSMA-CR, the join turn can be either a success, failure or it can be unused. Once every cycle - stations compare the current queue size with the previous cycle's queue size. If there is no difference between the previous cycle queue size and this cycle queue size,

the joining slot was either empty or a collision took place. A successful join slot results in an increase in the queue size by one. In real life, stations equipped with carrier sensing have ways to detect "start transmission" (beginning to carrier sense transmission) and "receive" (end of carrier sense). If a node starts to receive a message but cannot decode the packet, this means that the message was corrupted due to collision. The H_{MAX} in the subsequent join period can be computed based on a join slot's success, failure, or emptiness.

Every turn, nodes in the transmission group send out their data packets in their reserved slot. In the header of their data packets is a field called "current turn." Whenever an unjoined node has data to send, it can save the value of the current turn field of the most recent data packet locally; we call this the "backoff exit turn." When it comes time to send the actual data packet, the node references the backoff exit turn it saved previously. Additionally, when there has been a failure - it enters exponential backoff and again saves the turn that it exited backoff. The formula for the actual backoff based on the backoff exit turn, B, is in equation 3.1. Backoff exit turn is critical in this protocol because we use it to assign priority. Once the node has joined the queue, this state does not need to be updated or referenced.

In equation 3.1, B is the backoff exit turn. We use the term "backoff exit" generously because the turn where the node had data to send is technically the zeroth backoff exit. If a failure occurs, the node enters and then exits the first exponential backoff. QS is the size of the queue at the time that is readily available in one of the header fields of every packet. The first fraction $\frac{B}{QS}$ represents how early on in the queue cycle the exit event took place. As discussed previously, H_{MAX} is the variable that changes depending on the traffic at a given time. In times of heavy traffic, we might have a higher H_{MAX}, therefore multiplying the

range of discrete backoffs that we have and decreasing the probability of failure even further. Finally, τ is just an MCAT or transmission delay in the channel. τ is the perfect time between the start of signaling packet transmissions because a node scheduled to send at $x\tau$ will always be able to carrier sense an ongoing transmission at $(x-1)\tau$. After all, it takes τ seconds for any message to be carrier sensed by all other nodes.

$$T = \lfloor \frac{B}{QS} \times H_{Max} \rfloor \times \tau \tag{3.1}$$

One significant aspect of QSMA-CR is that the probability of success is not constant like other contention-free MAC protocols. With every failure, the distributed contention resolution algorithm decreases the probability of failure. Similarly, with ideal use of the join slot - empty or successful - the probability of failure increases to shorten the join slot and preserve maximum bandwidth for data transmissions. We have upper and lower limits for the H_{MAX} and the probability of a failed join, but it is not constant. When the queue size is small, we have another probability of failure for this case because nodes default to drawing from a uniform random distribution instead of when the backoff exit event took place. Using our 10 minutes 50 node experiment as an example: H_{MAX} briefly reached a global maximum of 20 during the beginning when contention was highest and reduced to the minimum allowable value of 4 for the rest of the experiment. As the contention reduced (due to the exponential backoffs and successful joins), only four mini-slots were required to resolve contention between the nodes. The contention resolution algorithm successfully kept the probability of failure as low during high contention. It also kept the joining period short otherwise to preserve maximum bandwidth for data transmissions.

Chapter 4

Performance Comparison

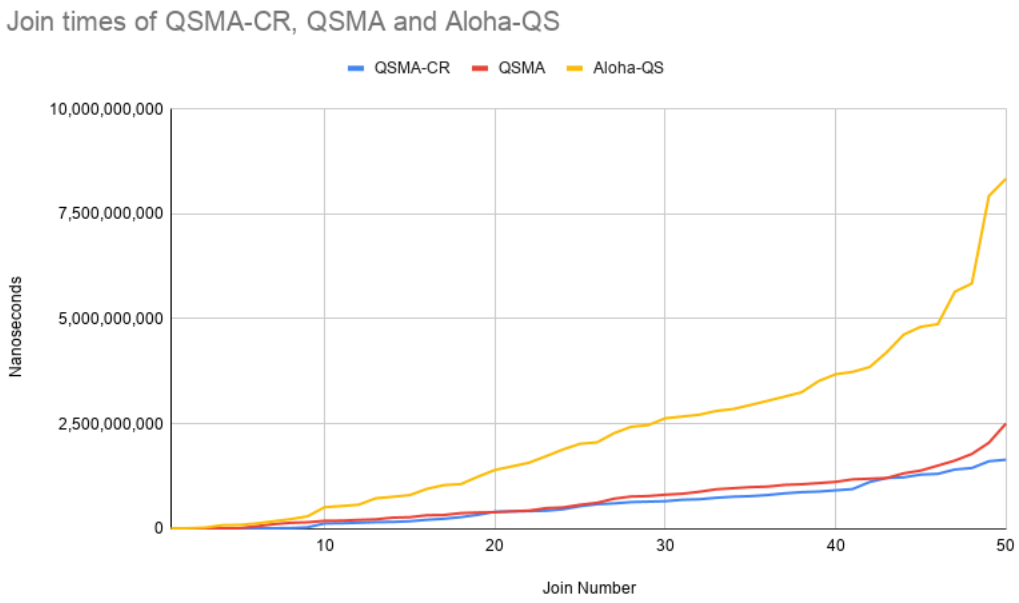

Figure 4.1: Plot of join times in nanoseconds for the first 50 joins for QSMA-CR, QSMA, and ALOHA-QS

This thesis uses Network Simulator 3, NS3, to simulate QSMA-CR, QSMA, and ALOHA-QS for 10 minutes with 50 stations. This scenario is very similar to the experiments on QSMA. [4] The reason we would want to evaluate the protocol

Join Number	QSMA-CR	QSMA	ALOHA-QS
1	1,396,750	984,971	11,874,490
2	1,769,990	1,357,101	13,270,072
3	3,662,367	7,013,657	28,213,444
4	6,304,118	9,642,141	85,214,370
5	8,007,391	13,014,861	90,992,161
6	10,073,209	59,975,173	126,867,690
7	12,509,854	108,208,186	178,290,207
8	15,320,152	141,649,979	228,117,116
9	34,420,045	154,157,695	296,281,848
10	126,775,905	189,249,793	513,140,721
11	130,704,962	193,170,748	541,444,528
12	143,500,059	210,341,420	572,539,327
13	157,527,193	224,337,536	725,033,614
14	162,575,136	264,443,541	761,709,216
15	178,842,157	275,263,026	801,175,888
16	213,396,657	321,323,252	949,082,546
17	238,070,941	327,477,624	1,039,182,309
18	276,978,071	372,775,927	1,063,101,378
19	332,297,878	386,573,183	1,240,317,976
20	404,746,024	393,843,170	1,400,589,488
21	412,406,994	416,384,209	1,484,910,565
22	420,440,969	432,220,709	1,573,417,443
23	428,848,616	490,738,142	1,727,908,126
24	463,972,901	508,063,850	1,889,377,402
25	536,723,106	571,408,008	2,024,133,429

Table 4.1: First 25 Join Times QSMA-CR vs QSMA vs ALOHA-QS

similarly to QSMA is that QSMA-CR extends upon it with more carrier sensing. We assume a fully-connected topology of 50 nodes that always have data packets to send. All nodes are within 300m of each other, resulting in maximum propagation delay, τ, of 1415 ns. We also assume no channel capture or channel error. The MAC data rate is 10 Mbps, and the transmission rate for the PLCP (Physical Layer Convergence Procedure) preamble and header of 24 bytes is 1 Mbps in the three protocols. All the three protocols in our scenario use a binary exponential backoff scheme with a maximum backoff of 256 epochs, where each epoch lasts

Join Number	QSMA-CR	QSMA	ALOHA-QS
26	584,356,495	618,735,455	2,059,216,513
27	604,155,471	716,726,588	2,278,098,054
28	634,862,434	767,973,436	2,429,599,541
29	645,508,294	778,594,276	2,468,870,400
30	656,528,079	811,379,940	2,631,539,444
31	690,704,709	834,111,288	2,673,601,238
32	702,469,343	880,674,718	2,717,057,856
33	738,759,919	940,837,961	2,806,765,048
34	763,647,679	965,608,681	2,853,012,016
35	776,531,598	991,318,327	2,948,300,930
36	802,895,234	1,004,544,825	3,046,381,254
37	843,623,830	1,045,148,771	3,147,252,419
38	871,462,566	1,059,120,924	3,250,913,481
39	885,839,569	1,087,614,749	3,517,049,484
40	915,153,563	1,116,852,239	3,680,918,392
41	945,207,371	1,176,622,331	3,736,935,760
42	1,113,942,995	1,192,082,524	3,851,762,835
43	1,208,589,328	1,207,914,701	4,204,620,242
44	1,224,830,827	1,320,378,441	4,626,053,721
45	1,290,715,297	1,386,103,782	4,810,852,700
46	1,307,703,059	1,503,583,074	4,873,846,598
47	1,411,107,313	1,623,668,273	5,646,540,572
48	1,446,382,443	1,781,357,086	5,843,897,969
49	1,607,806,612	2,049,637,686	7,926,538,315
50	1,644,573,294	2,505,953,385	8,338,000,217

Table 4.2: Last 25 Join Times QSMA-CR vs QSMA vs ALOHA-QS

100 μ s. We measured the throughput and the delay to join the queue - the time taken for a node to enter the QUEUE state. The size of each QSMA-CR header is 6 bytes: sender id, Q (size of the queue), E (entry turn), D (data ending), A (acknowledgment bit), and S (number of slots). The QSMA and ALOHA-QS headers skip the number of slots field and are 5 bytes in size. The signaling packets do not carry any payload, making their size 6 bytes for QSMA-CR and 5 bytes for QSMA and ALOHA-QS. The data payloads themselves of 218 bytes which is the same as a typical VoIP frame.

The main experiment that we ran between ALOHA-QS, QSMA, and QSMA-CR consists of 50 nodes and runs for 10 minutes. All the nodes intend to simultaneously join the queue causing high contention in the first few joins periods. This H_{MAX} increased to a maximum of 20 briefly and then settled at four for the rest of the experiment. The contention resolution algorithm enabled the joins to happen sooner. In tables 4.1 and 4.2, we can see that the join times in QSMA-CR are a lot lesser than ALOHA-QS and even QSMA. QSMA-CR does not compromise on the high bandwidth guarantees either and maintains an almost identical throughput as QSMA. Additionally, the total data transferred by all the nodes combined is higher because the joins happen sooner. The throughput measurement does not take the joining delay into account. Our contention resolution algorithm has done well in handling the contention throughout the entire experiment. Lesser latency alongside bandwidth guarantees are desirable in many real-world use cases, and this type of protocol would be beneficial for those applications.

One observation of note is in figure 4.1 is - QSMA-CR has more of a linear curve in the join times than QSMA and ALOHA-QS. ALOHA-QS has a quadratic curve, and QSMA's last few nodes make it slightly less linear than QSMA-CR. The QSMA-CR curve tells us that the join times grow relatively linearly with respect to the number of nodes in the queue, which means that no matter how big the queue is, the nodes can join with low latency. QSMA-CR is a more stable protocol that supports many nodes with ease. One of the main reasons for this is that our contention resolution algorithm never lets the exponential backoff reach exceedingly high backoffs, as seen in our experiments with QSMA (without collision resolution) and ALOHA-QS.

Given the same task, QSMA-CR achieves a throughput of 0.9945 and QSMA 0.9947. Contention causes the join slots to be longer. Our contention resolution

algorithm increased the maximum random timeout to 20 times the transmission delay or 20τ. Once more successful and empty join slots took place, the random maximum random timeout reduced to 4 times the transmission delay or 4τ, which is the length of every join slot in QSMA. The raw throughput values might be higher for QSMA; there is a valid reason for this. Firstly, our throughput measurements do not consider the joining delay. Here, the throughput is the amount of data transferred during the time that it was in the transmission group. The total data transferred throughout the experiment by all the nodes combined is still higher in QSMA-CR. If we calculated our throughput throughout the experiment's ten minutes instead, we might see different results. Secondly, QSMA-CR reaches the target queue size and joins all the nodes sooner. A larger queue size means that the throughput is lower because each cycle is longer.

Chapter 5

Future Work

A second experiment run on this protocol would be very illuminating. In our experiment, all fifty nodes attempt to join the transmission queue simultaneously. An alternative would be an experiment where nodes join the queue, send some data, and then exit the queue. In an experiment like this, QSMA-CR would have higher throughput because the effect that the join has on the throughput would be multiplied by the number of joins that each node has to make. For example, if 50 nodes join five times each, this gives us 250 joins in all. Each of these 250 joins would be faster in QSMA-CR, leading to a higher throughput. Additionally, this experiment is more realistic and would provide more insight into how the protocols react to this type of traffic. It might lead to more exciting avenues of research.

The exponential backoff mechanism in ALOHA-QS, QSMA, and QSMA-CR can be changed to better suit this protocol. The exponential backoff in ALOHA-QS was designed to exclude the backed-off node from contention for the next join or the next few joins. In QSMA-CR, we might want all nodes to be in contention during every join slot, even if they had a collision or failure in the previous join turn. With a high enough maximum mini-slots, H_{MAX}, it would be possible to

accommodate many nodes in the contention period. However, this again would come at the cost of throughput even though nodes would join sooner. Instead of having backoff exit every cycle as previously suggested, maybe back off could expire after two turns. Depending on the type of experiment and the payloads, these strategies might yield different, interesting results.

Bibliography

[1] Norman Abramson. Development of the alohanet. *IEEE transactions on Information Theory*, 31(2):119–123, 1985.

[2] R. Garces and J. J. Garcia-Luna-Aceves. Collision avoidance and resolution multiple access with transmission groups. In *Proceedings of INFOCOM '97*, volume 1, pages 134–142 vol.1, 1997.

[3] J. J. Garcia-Luna-Aceves, D. Cirimelli-Low, and N. Mashhadi. Aloha with queue sharing. In *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 174–183, Dec 2020.

[4] J.J. Garcia-Luna-Aceves and Dylan Cirimelli-Low. Queue-sharing multiple access. In *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '20, page 93–102, New York, NY, USA, 2020. Association for Computing Machinery.

[5] JJ Garcia-Luna-Aceves and Ashok N Masilamani. Nomad: Deterministic collision-free channel access with channel reuse in wireless networks. In *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 1–9. IEEE, 2011.

[6] Gentian Jakllari, Mike Neufeld, and Ram Ramanathan. A framework for frameless tdma using slot chains. In *2012 IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS 2012)*, pages 56–64. IEEE, 2012.

[7] Gentian Jakllari and Ram Ramanathan. A sync-less time-divided mac protocol for mobile ad-hoc networks. In *MILCOM 2009-2009 IEEE Military Communications Conference*, pages 1–7. IEEE, 2009.

[8] Andrew Muir and JJ Garcia-Luna-Aceves. An efficient packet sensing mac protocol for wireless networks. *Mobile Networks and Applications*, 3(2):221–234, 1998.

[9] Lawrence G Roberts. Aloha packet system with and without slots and capture. *ACM SIGCOMM Computer Communication Review*, 5(2):28–42, 1975.

[10] Aggeliki Sgora, Dimitrios J Vergados, and Dimitrios D Vergados. A survey of tdma scheduling schemes in wireless multihop networks. *ACM Computing Surveys (CSUR)*, 47(3):1–39, 2015.

[11] Dimitrios J Vergados, Natalia Amelina, Yuming Jiang, Katina Kralevska, and Oleg Granichin. Toward optimal distributed node scheduling in a multihop wireless network through local voting. *IEEE Transactions on Wireless Communications*, 17(1):400–414, 2017.

[12] ZANG Wan-Yu YU Meng XIE and Li SUN Zhong-Xiu. A survey of on-demand routing protocols for ad hoc mobile networks [j]. *Chinese Journal of Computers*, 10, 2002.

[13] Wenxin Xu and Graham Campbell. A distributed queueing random access protocol for a broadcast channel. In *Conference Proceedings on Communications Architectures, Protocols and Applications*, SIGCOMM '93, page 270–278, New York, NY, USA, 1993. Association for Computing Machinery.