DEPARTMENT OF ELECTRICAL
AND ELECTRONICS
ENGINEERING

EE491 PROJECT REPORT

Predicting Number of Transmitting Nodes in WLAN

Berke EREN 250206008

Korkut Emre ARSLANTÜRK 250206039

Prof. Barış ATAKAN

DATE: 15/01/2022

ABSTRACT

802.11 protocol was standardized by the IEEE for WLANs(Wireless Local Area Networks). Distributed coordination function is the main medium access control method for 802.11. Distributed coordination function is a binary slotted exponential backoff carrier sense multiple access with collision avoidance (CSMA/CA) method. The Markov Chain approach is a rule in which the statement is determined independently of the previous statements. The Bianchi Model estimates the number of nodes using the distributed coordination function properties and the Markov Chain. In this study, the validation of the Bianchi Model has been executed. This model offers a high accuracy rate under ideal channel conditions and for a certain number of terminals. That model applies to both the basic access and RTS/CTS access modes used by DCF for packet delivery. In this study, the simulation results obtained using NS2 are tabulated and compared with the results calculated from the analytical Bianchi Model. According to the simulation results, the collision probability of the packages was recorded. Then, this data was used to estimate the node number using Gaussian Distribution. Finally, the estimated number of nodes was compared with the number of nodes where the simulation was performed, and it was observed that the accuracy of results was very high.

TABLE OF CONTENTS

ABBREVIATIONS

IEEE: Institute of Electrical and Electronics Engineers

WLANs: Wireless Local Area Networks

CSMA/CA: Carrier-Sense Multiple Access with Collision Avoidance

NS2: Network Simulator 2

DCF: Distributed Coordination Function

MAC: Media Access Control

PHY: Physical Layer

DIFS: Distributed Interframe Space

SIFS: Short Interframe Space

CW: Contention Window

ACK: Acknowledgement

RTS: Request to Send

CTS: Clear to Send

DSDV: Destination-Sequenced Distance Vector

UDP: User Datagram Protocol

TCP: Transmission Control Protocol

CBR: Cooperative-Distributed Routing

AGT: Agent Trace

RTR: Router Trace

IFQ: Interface Queue

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Many mobile devices today utilize several wireless interfaces. As a result, there is a research issue about device-to-device communication dedicated to benefiting the appropriate interface for communicating with other users directly, regardless of the network architecture. IEEE 802.11 is the de facto standard and able to determine medium access control and the physical layers for WLANs. IEEE 802.11 working groups are responsible for proposing and developing MAC and PHY layer WLAN specifications for mobile and portable stations [1].

The fundamental concepts of the IEEE 802.11 MAC layer will be briefly described, as they are required to understand performance evaluation. Although there are many different substrates, this study is based on 802.11b. To analyze the performance of the 802.11 protocol, one of the most well-known analytical models, the Bianchi Model [2], has been studied. Information is given about the Markov Chain [5] to understand this model. Finally, our simulation results and estimation results are compared. This study aims to validate the Bianchi Model by predicting the number of transmitting nodes in the WLAN.

In Network Simulator 2, wireless network simulations with different nodes are created according to the 802.11 protocol. The probability of collision is calculated according to the values in the trace file for simulations created on different node numbers using the Markov Chain rule. Subsequently, the number of nodes is estimated using Gaussian Distribution on the recorded values. According to the generated estimation algorithm, it is expected to obtain an output close to the actual number of transmission nodes.

2. PROBLEM DEFINITION

Mathematical calculations that need to be overcome to make a node prediction have been specified.

2.1. Wireless Local Area Network:

WLAN is a computer network that uses wireless communication, which is developed to connect two or more devices in a limited area to form a local area network and enables all the events that can be done in wired communication. This gives users the ability to move within the space and stay connected to the network. Via a gateway, a WLAN can also provide connectivity to the wider Internet.

2.1.1. IEEE 802.11 Protocol:

IEEE 802.11 is the de facto standard for an ever number of WLAN applications developed throughout the world. For WLANs, it specifies both the media access control and physical layers. IEEE 802.11 workgroups are responsible for proposing and developing MAC and PHY layer WLAN standards for mobile and portable stations. The MAC layer is placed above one of many possible physical layers in this standard [3].

2.1.2. Distributed Coordination Function:

DCF is a protocol that optimizes throughput while preventing packet collisions by combining carrier sensing with a four-way handshake. A packet collision occurs when a node receives several packets simultaneously, resulting in neither packet being successfully received. This collision avoidance environment access control substrate technique is used in regions where carrier-sense multi-access (CSMA/CA) is required.

The channel activity is monitored by a station with a new packet to transmit. The station transmits if the channel is idle for a period equal to a distributed interframe space (DIFS). Otherwise, if the channel is detected as busy, the station continues to monitor it until it is determined to be idle for a DIFS.

DCF uses a discrete-time backoff scale for efficiency reasons. The period immediately after an idle DIFS is slotted, and a station can only transmit at the beginning of each slot time. The slot time size (σ) is set to the amount of time it takes for any station to notice the transmission of a packet from another station. Since the protocol used in the project is the 802.11b version, it was based on the parameters in Table 1. The slot time was chosen accordingly.

Parameters	**Value**
SIFS	10 μs
DIFS	50 μs
Slot Time Duration	20 μs
Propagation Delay	2 μs
Physical Layer Header	192 bits / 1Mbps
Mac Header	224 bits / 11Mbps
RTS	160 bits / 1Mbps
CTS	112 bits / 1 Mbps
ACK	112 bits / 1 Mbps
CWmin / CWmax	32 / 1023

Table 1: 802.11b Parameters. [4]

The exponential backoff mechanism is used by distributed coordination function. The backoff time is uniformly chosen in the range for each packet transmission (0,w-1). The value is known as the contention window, and it is determined by the number of packet transfers that have failed. At the first transmission attempt, w, is set equal to a value CWmin called minimum contention window. After the failed transmission, the content window value is doubled. This value increase continues until the value of 2^nxCWmin. The values CWmin and CWmax reported in the final version of the standard are physical layer-specific and are summarized in Table 1.

Figure 1 shows an example WLAN with two stations, A and B share the same wireless channel. Station B waits for a DIFS at the end of packet transmission and chooses a back-off time of 8 before transmitting the next packet. Station A's first packet must arrive at the time indicated by the arrow in the figure, and it is transmitted after a DIFS. The transmission of packet A corresponds to a back-off value of 5 for station B in the middle of Slot Time. The backoff time is locked to 5 when the channel is detected as busy, and the backoff counter is only reduced when the channel is detected as busy.

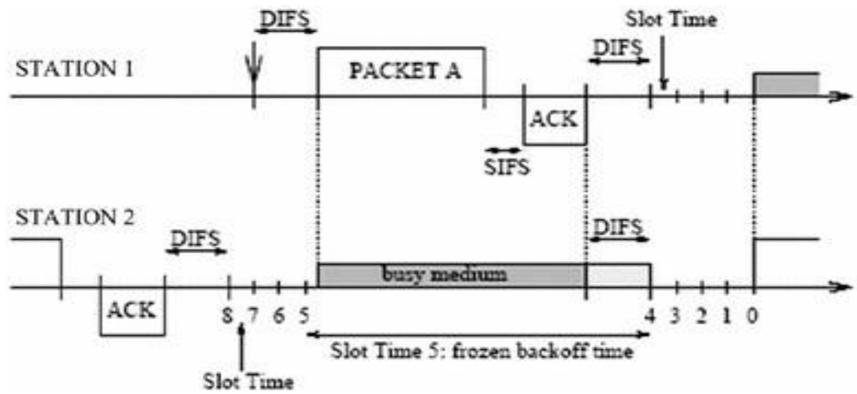

Figure 1: Example of a Basic Access Mechanism with Backoff Time [2]

Since this CSMA/CA protocol does not rely on the stations' capacity to detect a collision by hearing their own transmission, the destination station sends an ACK to signal successful packet reception. As shown in Figure 1, after a period called a short interframe space, the ACK is sent at the end of the packet. No other station can detect the channel idle for a DIFS until the end of the ACK because the SIFS is shorter than a DIFS. Suppose the transmitting station does not receive the ACK within a defined timeout or detects the transmission of a different packet on the channel. In that case, the packet transmission is rescheduled according to the backoff rules. The ACK signal is transmitted by the receiving station to the transmitting station to signify that it is ready to accept data. It's also used to make that the data sent was received without problems.

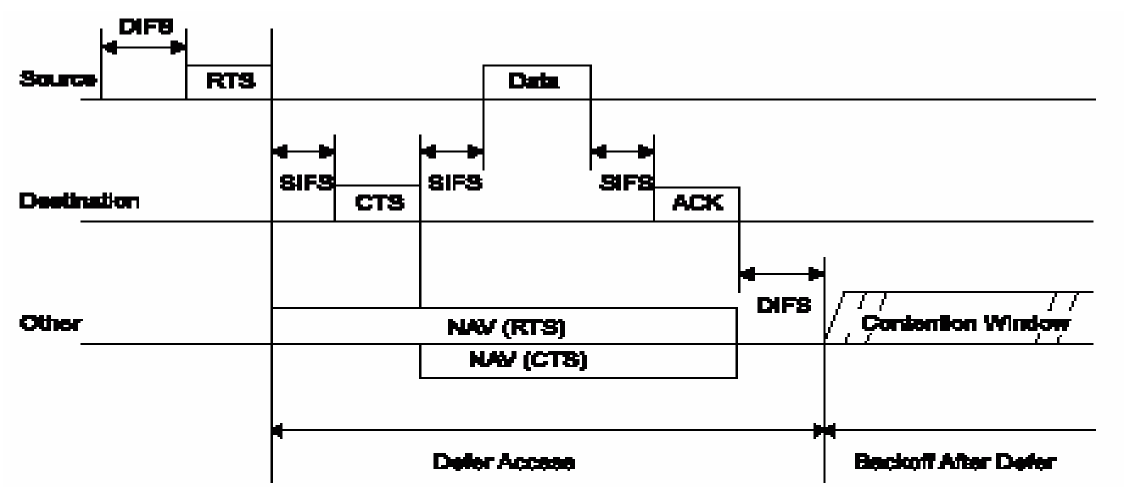

Figure 2: RTS/CTS Access Mechanism [3]

The basic access mechanism refers to the two-way handshaking approach mentioned for packet transmission. DCF specifies a fourth four-way handshaking mechanism that may be utilized for packet transport if desired. RTS/CTS is the term given to this process, seen in Figure 2. A station that wants to send a packet first waits for the channel to be detected idle for a DIFS. Then it obeys the recoil mentioned above rules and sends a particular short frame, which is called to request to send instead of the packet (RTS). When the receiving station detects an RTS frame, it sends a clear to send (CTS) frame following a SIFS. Only if the CTS frame is successfully received is the transmitting station allowed to send its packet. When a station is hidden from the transmitting or receiving station, it can successfully delay further transmission by detecting just one frame among the RTS and CTS frames, preventing a collision [2].

2.2. Markov Chain

Bianchi's Model calculates the probability of a packet transmission failure due to collision [2]. It accepts that the channel is in the ideal conditions like there is no hidden terminal and capture effect. The Markov Chain is a mathematical system that goes between transitions from one state to another based on probabilistic criteria. The transition probability matrix is shown in equation 1.

$$\mathrm{P(i,j)} \geq 0, \mathrm{i,j} \in \mathrm{X} \text{ and } \sum_j \mathrm{P(i,j)} = 1, \mathrm{i} \in \mathrm{X}. \quad (1)$$

If it is accepted that 'i' is the state, it will jump to 'j,' independently of its previous values. It can be shown like equation 2.

$$\mathrm{P}[\mathrm{X}_{n+1} = \mathrm{j} | \mathrm{X}_n = \mathrm{i}, \mathrm{X}_m, \mathrm{m} < \mathrm{n}] = \mathrm{P(i,j)}, \mathrm{i,j} \in \mathrm{X}, \mathrm{n} = 0,1,2 \ldots. \quad (2)$$

A critical study on the Wi-Fi MAC protocol has been presented by G. Bianchi. This analysis is based on the assumption that there are (n – 1) Wi-Fi devices and the Access Point, and they always have data to transmit.

The probabilities of the transitions in the Markov Chain are presented. In there, W_i and m are related to CW_{min} and CW_{max} values (according to 802.11b), as can be seen in equation 3.

$$W_i = 2^i(CW_{min} + 1) \; and \; CW_{max} + 1 = 2m(CW_{min} + 1). \quad (3)$$

To analyze the Markov Chain {s(k), b(k)} for a specified device, it is accepted that s(k) determines how many times the pending packet has been attempted to be transmitted before while b(k) specifies the back-off counter at the k's epoch.

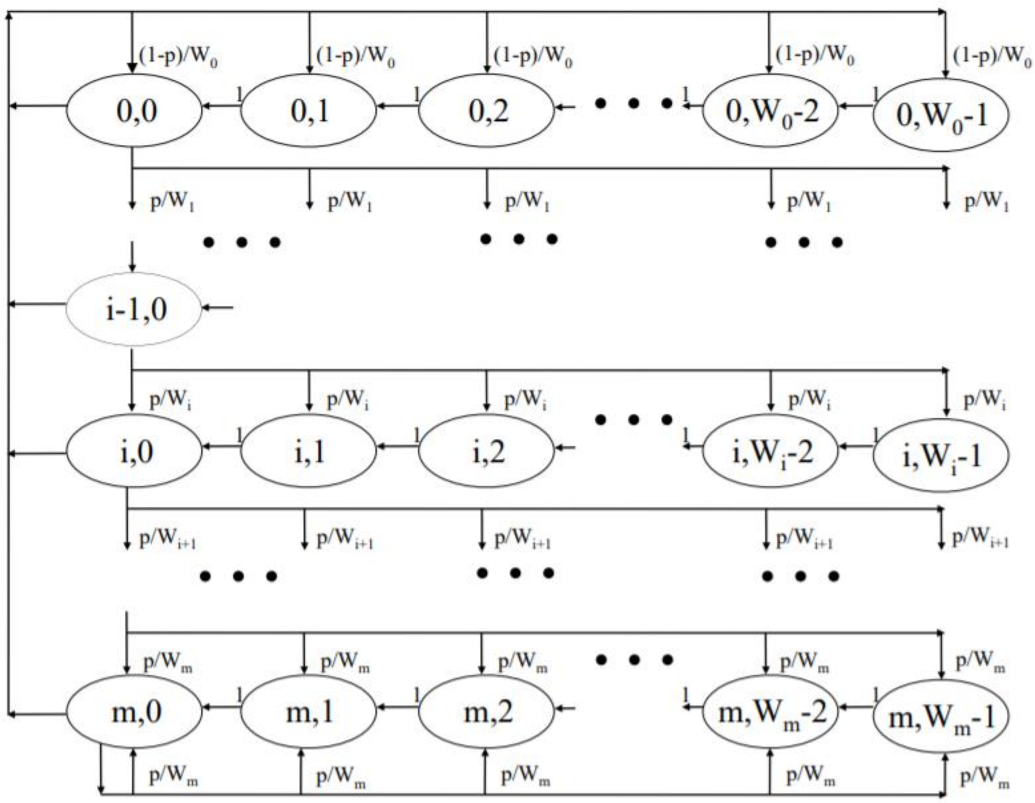

Figure 3: Backoff Window Size Markov Chain Transition Probabilities [5]

A station, which has a new packet to transmit, can be imagined in order to understand Figure 3. The node at the top of the figure represents the state of the station. The station computes a backoff value uniformly between 0-31 after waiting for DIFS.

At appropriate epochs, the station decrements its backoff counter. For example, the state could shift from (0, 5) to (0, 4). The backoff counter eventually resets to zero, resulting in a state of (0, 0). Thus, the station transmits. The station determines different backoff values, X, from 0 to 63 when the transmission collides. The station's status becomes to (1, X). In that shown, the first component defined one attempt is already done by the station. Whether the station goes up to the top or not is related to the success of transmission.

In that assumption, the station transmits probability 1 – p and collides probability equals p. Those probabilities are estimated independently from the state, station, and previous states. The likelihood of station attempts to transmit can be found by analyzing the Markov Chain.

The amount of transmitting attempts is shown by α. Furthermore, it is accepted that there are n stations, and they have the same average α. Moreover, their transmission attempts are independent. The probability of a station having successful transmission is equal to other n-1 stations that do not transmit. This α depends on probability p, as seen in equation 4.

$$1 - p = (1 - a(p))^{n-1} \tag{4}$$

Attempt probability can be calculated after solving for 'p'. The probability transition matrix indicated P while π distinguished the invariant distribution. Also, the fact is that π equals πP from the definition of the invariant distribution. Consequently, equation 5 can be written.

$$\pi(x) \sum_{y \neq x} P(x, y) = \sum_{y \neq x} \pi(y) P(y, x) for\ each\ x \tag{5}$$

Equation 6 may be found by applying the balanced equations and transition probabilities to the previous equation.

$$\pi(i - 1{,}0)p = \pi(i, 0) \rightarrow \pi(i, 0) = p^i \pi(0{,}0) for\ 0 < i < m. \tag{6}$$

Equations 7 and 8 can be expressed as follows by recursively applying the balanced equations and some algebraic manipulations.

$$\pi(m - 1{,}0)p = (1 - p)\pi(m, 0) \rightarrow \pi(m, 0) = \frac{p^m}{1-\mathrm{p}} \pi(0{,}0) \tag{7}$$

$$\pi(i, k) = \frac{W_i - k}{W_i} \pi(\mathrm{I},0) \text{ for i } \in \{0, \ldots, \mathrm{m}\} \text{and k } \in \{0, \ldots, w_{i-1}\}. \tag{8}$$

An expression for every π(i, k) can be determined by employing previous equations. Finally, using summing property, transition probability can be found as can be seen Equation 9[5]. In that shown W equals to CWmin + 1.

$$a(p) = \frac{2(1-2p)}{(1-2\mathrm{p})(\mathrm{W}+1)+\mathrm{p}\mathrm{W}(1-(2p)^m} \tag{9}$$

3. PROPOSED SOLUTION

Taking into account the calculations defined in the previous part, the necessary data were obtained as a result of the simulation, and the prediction process was performed with the algorithms created.

3.1. Simulation Process

To obtain the required values in the Markov Chain, simulations were performed in Network Simulator 2 at different node numbers and at different times.

3.1.1. Network Simulator 2:

Network Simulation 2, or NS2, is an essential event-driven simulation tool that has proven helpful in studying the dynamic nature of communication networks. Both wired and wireless network services and protocols may be simulated using NS2. NS2 allows users to specify network protocols and simulate their associated behaviors in general [6].

```
set val(chan)           Channel/WirelessChannel
set val(prop)           Propagation/TwoRayGround
set val(netif)          Phy/WirelessPhy
set val(mac)            Mac/802_11
set val(ifq)            Queue/DropTail/PriQueue
set val(ll)             LL
set val(ant)            Antenna/OmniAntenna
set val(ifqlen)         50
set val(nn)             3
set val(rp)             DSDV
set val(x)              500
set val(y)              400
```

Figure 4: Define Options

In these commands in Figure 4, some basic parameters for simulations are specified, and information is given for different layers. To explain some of these commands:

- Using the vlan (chan) command, the channel between the nodes created in the topology is provided to be a wireless feature.
- It has been determined that the propagation on the channel by vlan (prop) is a two-beam ground reflection model. In this way, both direct and ground reflection is used.
- The queue type created with vlan (if) manages how packets are buffered while waiting to be forwarded. In the Drop-Tail mechanism used, when the queue is full to its maximum

capacity, new incoming packets are dropped until the queue has enough space to accept incoming traffic.

- The DSDV protocol, created by the vlan (rp) command, is a hop-on-hop vector routing system that requires each node to regularly publish routing updates.

```
$ns node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
                 -ifqLen $val(ifqlen) \
                 -antType $val(ant) \
                 -propType $val(prop) \
                 -phyType $val(netif) \
                 -channelType $val(chan) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace OFF \
                 -macTrace ON \
                 -movementTrace OF
```

Figure 5: Configure the Nodes

In Figure 5, it is determined what kind of protocols should be applied in the node configuration. In this project, the AgentTrace and MacTrace nodes are set to open. This is due to the fact that we want to include only the collision, received, and transmitted information in the trace file. With AgentTrace, Agents are used to separate protocol statuses from nodes. MacTrace takes care of monitoring MAC protocol packets. Whether or not the MacTrace configuration is turned on directly affects the results. This is because this command decides whether the RTS and CTS protocols should be applied or not. It is observed that the collisions that will be obtained if it is open are reduced.

The protocol layer just above the Internet Layer is known as the Host-to-Host Transport Layer and is also layer 4 of the OSI standards. There are two commonly used protocols in the Transport Layer. These are:

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP).

Both protocols provide data transmission and control between the Application Layer and Internet Layer.

```
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(3) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(0) $null_(0)
$ns_ connect $udp_(0) $null_(0)

set udp_(1) [new Agent/UDP]
$ns_ attach-agent $node_(3) $udp_(1)
set null_(1) [new Agent/Null]
$ns_ attach-agent $node_(1) $null_(1)
$ns_ connect $udp_(1) $null_(1)

set udp_(2) [new Agent/UDP]
$ns_ attach-agent $node_(3) $udp_(2)
set null_(2) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(2)
$ns_ connect $udp_(2) $null_(2)
```

```
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) attach-agent $udp_(0)
$ns_ at 0.1 "$cbr_(0) start"

set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) attach-agent $udp_(1)
$ns_ at 0.1 "$cbr_(1) start"

set cbr_(2) [new Application/Traffic/CBR]
$cbr_(2) attach-agent $udp_(2)
$ns_ at 0.1 "$cbr_(2) start"
```

Figure 6: Setting of UDP Connection

UDP connection is provided in Figure 6. UDP is a transport layer communication protocol. Because it is a connectionless communication protocol, no connection needs to be established before sending data packets. Here, Agent is the sending node. Null is the receiving node. The agent node is permanently fixed and is the access point in the topology. The access point is a node that creates a wireless local area network. The CBR parameter is set to regulate the data traffic. CBR is set to generate 1 Kbyte packet at 100 packets per second.

The primary purpose of using UDP instead of TCP in the project is that UDP does not guarantee the delivery of data to the recipient and does not retransmit lost packets. Thus, it performs faster data transfer than TCP. Another important criterion is that as a result of missing packages not being sent again, more packages can be moved, and the error rate increases. Thus, the result we will get will be more optimized.

3.2. Use of Simulation Data

Different parameters obtained from the Network Simulator 2 are saved in the trace file. On trace files, each block stores different information. A line from the output trace as an example is:

```
D 0.295025000 _2_ IFQ --- 105 cbr 210 [0 1 2 800] ------- [2:1 1:0 32 0] [52] 0 0
```

In that output, the first place determines the situation such as “received", “sent", “forwarded" and “dropped" with a letter. The time of simulation is specified in the second place. The third position provides information on the node number.

MAC, AGT, RTR, or IFQ will take fourth place, referring to the MAC layer, transport layer (such as TCP), routed packet, or events owing to the interference priority queue (such as packet drops), respectively. After the dashes, the packet's global sequence number may be seen in the fifth position. The sixth field contains information about the type of packet. The packet size in bytes is shown in the seventh position. The following field contains information about the Mac layer. The hexadecimal value in that location indicates the time it will take to transfer a data packet across the wireless channel. In addition, the second number in that bracket gives the MAC-id of the sending node, whereas the third number specifies the MAC-id of the receiving node. The MAC type is determined by the last integer in the bracket, ETHERTYPE IP. The following field, which contains integers in square brackets, contains information on the packet's IP source, destination addresses, and time to live. The last field contains TCP information (sequence number and acknowledgment number) [7].

Event (r,s,f or D)	Time	Node Number	Refers to Layer	global sequence number of the packet	packet type	packet size	mac layer information	IP source, destination addresses and ttl (Time To Live) of the packed.	Tcp Information

Figure 7: Trace File Format of 802.11b

Event information on the trace file is considered to apply the Markov Chain. The amount of drop events and total event values on the trace file were calculated using AWK programming. The number of dropped divided by the number of all events gives the collision ratio, and the calculation of that ratio is shown in Figure 8.

```
BEGIN{
Dpacket= 0
spacket=0
rpacket=0
fpacket=0
total= 0
}
{
event = $1

if(event=="D"){
Dpacket++;
}

if(event=="r" || event="s" || event="D"|| event="f"){
total++;
}

}

END{
printf("d packets are %d \n", Dpacket);
printf("total packets are %d \n", total);
printf("ratio are %f \n", ((Dpacket)/total));
}
```

Figure 8: Defined Algorithm to Calculate Collision Probability

Also, the collision probability value is calculated according to the Markov Chain to compare. Finally, intervals are determined according to the standard deviation and the collision probability values. Estimation is performed with Gaussian distribution using the simulation results in the algorithm created with the values found from Markov Chain on Python.

```
j=2

while(j<20):

    i=2;
    while (i<20):
            if(t_fivems[j-2]<x_data[i-2]+(var_for) and t_fivems[j-2]>x_data[i-2]-(var_for)):
                result=i
                print(result)
                break
            else:
                i=i+1;
    j=j+1;
```

Figure 9: Estimation of Node Number Algorithm

In this code, because up to 20 nodes are estimated, the while loop continues as long as it provides the j is more diminutive than twenty condition. In the If statement, it is checked whether the result obtained from the simulation is between the standard deviation and the sum and difference of the calculated result from the Markov Chain or not. If the result is not in this interval, the other iteration is switched to. If the result is in this interval, the prediction is suppressed.

4. RESULTS AND DISCUSSIONS

Collision probabilities and node estimations according to the different simulation times can be seen in the tables.

	Collision Probability				
		Observed Results According to Simulation			
Node Number	Calculated using Markov Chain	25ms	50ms	100ms	125ms
3	0.105	0.072717	0.073041	0.073196	0.073222
4	0.145	0.138020	0.138321	0.138468	0.138489
5	0.18	0.182684	0.182970	0.183103	0.183123
6	0.208	0.218308	0.219171	0.219604	0.219746
7	0.2315	0.244756	0.246369	0.247074	0.247245
8	0.2528	0.279832	0.286449	0.289835	0.290447
9	0.272	0.298623	0.304325	0.307220	0.307751
10	0.28	0.312299	0.318268	0.321205	0.321735
11	0.289	0.316774	0.317998	0.318614	0.318713
12	0.3035	0.328786	0.329888	0.331084	0.331343
13	0.3165	0.342604	0.351873	0.356581	0.357469
14	0.329	0.352657	0.358931	0.362100	0.362785
15	0.34	0.353314	0.356095	0.357564	0.357850
16	0.3502	0.370320	0.374763	0.377173	0.377619
17	0.369	0.376126	0.380656	0.383113	0.383502
18	0.377	0.377583	0.383995	0.388003	0.389154
19	0.385	0.386096	0.390603	0.393109	0.393544
20	0.3917	0.391884	0.395802	0.397954	0.398344

Table 2: Comparison of Collision Probability

In Table 2, the collision probabilities are calculated according to the different node numbers using the Markov Chain. After that, simulations were performed for different time values for the same node numbers. It has been observed that the calculated collision probability values from the Bianchi Model and the observed collision probability values as a result of simulation coincide with each other. It has been found that changes in simulation times sometimes bring the obtained result closer to the calculated value and sometimes take it away. This is due to the fact that as a result of an increase in the number of nodes, new node locations in the topology are randomly determined, and changes in these locations significantly change the outcome. Due to the storage space of the device used, simulations longer than 125ms could not be performed.

	Estimation of Node Numbers According to Simulation Results			
Node Number	**25ms**	**50ms**	**100ms**	**125ms**
3	2	2	2	2
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	9	9	10	10
9	10	11	12	12
10	11	12	12	13
11	12	12	12	12
12	12	13	13	13
13	13	15	15	15
14	14	16	16	16
15	14	15	15	16
16	16	17	17	17
17	16	17	17	17
18	17	17	18	18
19	17	18	18	18
20	18	19	19	19

Table 3: Node Estimation

The node estimation was performed based on the Markov Chain as specified Equations 4 and 9, using the collision probability value obtained from the simulation results. The prediction results showed slight differences according to the simulation time. Since an optimal result is not obtained when obtaining simulation results, some estimation values vary depending on the time change. When performing this estimation, the gaussian distribution method was used in Python, as mentioned in the problem-solution section.

		Without RTS/CTS Simulation Results at 25ms	
Node Number	Calculated Col. Prob. using Markov Chain	Col. Prob	Node Est.
3	0.105	0.06	6
4	0.145	0.105	6
5	0.18	0.145	7
6	0.208	0.18	7
7	0.2315	0.208	8
8	0.2528	0.2315	8
9	0.272	0.2528	8
10	0.28	0.272	8
11	0.289	0.28	10
12	0.3035	0.289	8
13	0.3165	0.3035	8
14	0.329	0.3165	10
15	0.34	0.329	10
16	0.3502	0.34	10
17	0.369	0.3502	12
18	0.377	0.369	10
19	0.385	0.377	11
20	0.3917	0.385	12

Table 4: Simulation Results without RTS/CTS

The RTS/CTS scheme designed in the 802.11 protocol is suitable for combatting the hidden terminals problem that occurs when mobile station pairs cannot hear each other. Therefore, RTS/CTS is turned on, and its effect on accuracy can be seen in the table. It can be observed that the simulation results are close to the expected value. It has been observed that the collision probability changes as the simulation time changes. Since this change was minimal, it did not affect the estimation.

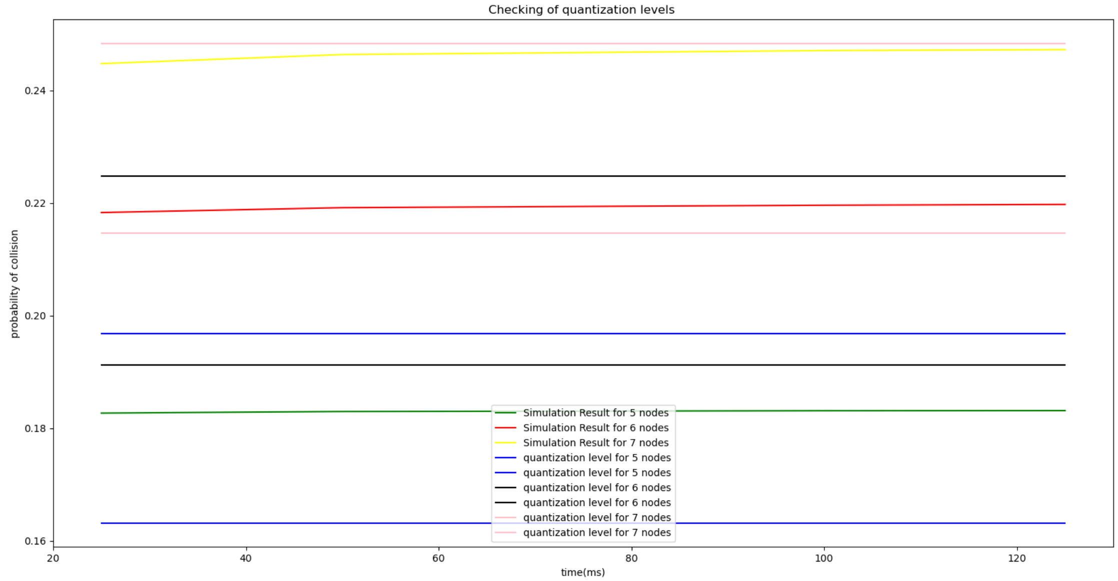

Figure 10: Interval Control of Simulation Results at 5,6,7 Nodes

The obtained results and the expected node numbers are consistent. For nodes 5,6 and 7, all of the simulation results are in the specified interval determined for estimation, as shown in Figure 9. When calculating the quantization level, the standard deviation of the values was taken and the collision probability value from the formula for the same node was assigned as mean.

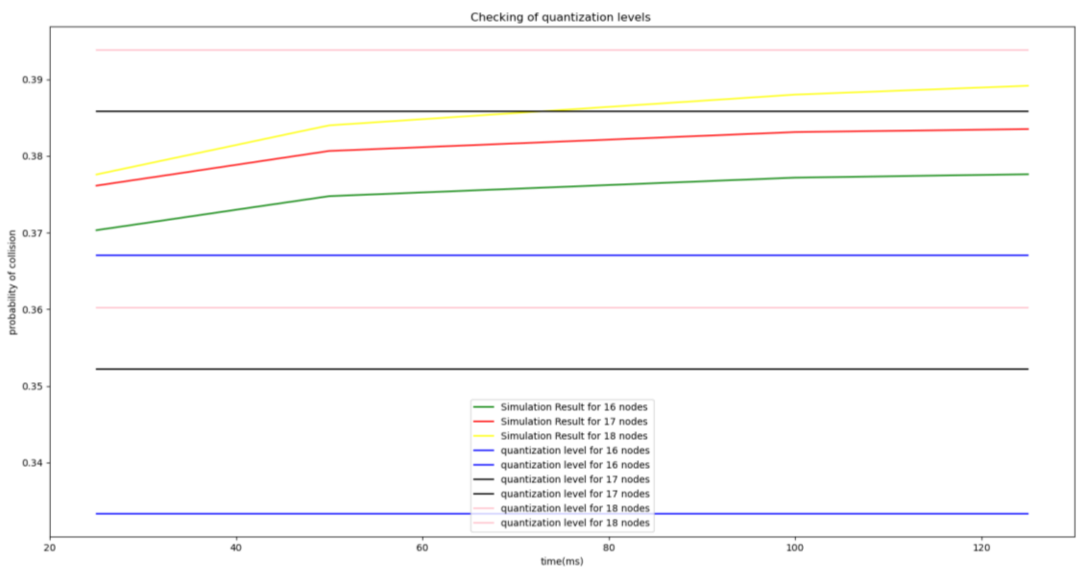

Figure 11: Interval Control of Simulation Results at 16,17,18 Nodes

As seen in Figure 10, the wrong estimation occurred for 16 nodes because of exceeding the determined levels. As a result of parameters such as the positions of the added nodes or the simulation time, deviations from the estimate may occur. Location detailed location detection can be done to get more accuracy.

5. CONCLUSION

It was aimed to verify the Bianchi Model by estimating the number of transmitter nodes in the WLAN. Basically, literature research was conducted on the implementation of the 802.11 b Protocol, the Markov Chain, and the Bianchi Model. NS2 simulation was used to measure the parameters that needed to be obtained for Markov Chain. AWK code was written to calculate the obtained simulation result values automatically. Then, the necessary operations were performed via Python, and node estimation was performed. By interpreting the received values, it was determined that there are consistent results, but there are cases that can be improved for closer results. Thus, the project was completed using many applications and methods.

In conclusion, the algorithm for estimating the number of nodes, based on the Bianchi Model in this project, provides consistent results. Inconsistent results for some number of nodes are related to the ideality of the conditions such as location and time. In the topology created in the simulation, each node has a different location in the XY plane. When the position change, some changes in the collision rate are observed. In particular, the collision rate decreases as the nodes get closer to each other. The lack of a standard for this phenomenon causes differences in the results obtained. Another issue is the simulation time variability. When the simulation time changes, the collision rates vary. However, since the recording space on the virtual machine of the computer used was not sufficient, it could not be simulated for high times.

During this entire project, the effects of teamwork were observed. In addition, the experience was gained on the progress of academic work and the acquisition of knowledge related to this work. The results of the literature review and academic article review were obtained. In addition to these, significant changes in career planning have been realized thanks to this project.

REFERENCES

[1] M. H. Manshaei and J.-P. Hubaux, “Performance Analysis of the IEEE 802.11 Distributed Coordination Function: Bianchi Model,” Mar. 2010.

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

[3] L. S. Committee, ANSI/IEEE Std 802.11:Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Computer Society, 1999.

[4] N. Gupta and C. S. Rai, “A simple mathematical model for performance evaluation of finite buffer size nodes in non-saturated IEEE 802.11 DCF in ad hoc networks,” Advances in Intelligent Systems and Computing, pp. 505–512, 2015.

[5] Walrand, J., & Parekh, S. P. “*Communication Networks: A concise introduction*.” Morgan & Claypool. 2018.

[6] T. Issariyakul and E. Hossain, “Introduction to network simulator 2 (NS2),” *Introduction to Network Simulator NS2*, pp. 21–40, 2011.

[7] Altman, E., & Jiménez Tania. (2012). *Ns Simulator for beginners*. Morgan & Claypool.