Report of The 802.11 Mac Protocol

Bolong Zhang & Bowen Li

1.Abstract

Implement simulation programs to evaluate Wifi 802.11 MAC protocols including DCF and RTS/CTS.Get better understanding of the MAC protocols .
This project include 2 programs. In the first program, we will use Mersenne Twister algorithm to generate a random traffic file and seed to make sure weather the result is random or constant. In the second program, we will use 2 different MAC protocols:DCF, which includes using acknowledgement, random wait, virtual sensing, and binary exponential bake-off. RTS/CTS ,which bases on DCF and adds RTS/CTS.
Lastly, drawing the conclusion about the throughput/latency in these 2 protocol.
We will code this program by C++ language and run it at Linux OS.

2. Generating the traffic file

Using traffic generator to generate the traffic files. The traffic generator should execute with the following parameters

./traffic_generator num_node average pkt_size offered_load num_pkts_per_node method [seed]

num_node: the number of stations in the system that can send and receive packets. The value of this parameter is from 2 to 200.

average pkt_size: the average packet size in bits. The value of this parameter is from 100 to 2000.

offered_load: it value is from 0.01 to 10.

pkts_per_node: the number of packets sent from each station in the traffic file.

gap: after the station sent a packet, we should select a random time t form [0, 2*gap], then this station will send the next packet after time t.
pkt_size/(pkt_size+gap)=offered_load/num_node

seed: It is optional. This parameter that is for setting the seed for random number generator. If this parameter is not provided, the program should generate a different traffic with the same statistical characteristics each time it runs. If the same seed is provided, the program should generate exactly the same file every time.

method: In this project, the packet size is different with each packet. So we should use two different probability distributions exponential distribution, uniform distribution to generate the packet size. ***It's value is exp and uniform.***

Then using the Mersenne Twister algorithm to generate the synthetic random traffic and store the traffic in a file.

The first line of the traffic file contains a number telling the total number of packets in the file. After that, the traffic file contains lines of the formats :
pkt_id src_node dst_node pkt_size time
and are sorted based on the time. Each line represents a packet of size pkt_size in bits from src_node to dst_node that is ready to be sent at time time. The pkt_id must be unique for each packet in the file.

3. Using MAC protocols to calculate the throughout

3.1 DCF Protocol

Each station wait random time to send the packet.

Each station can detect if the frame it sent collided with frames from other stations. If yes, choose a random time from contention window and resend again.

pseudocode:

```
Read a Packet
Wait a DIFS+ Random time from[0,15]slots
If channel is free, send the packet, the channel is busy. After sending the packet, wait DIFS and wait to get the acknowledgement, the channel is free again.
It channel is busy when the node is waiting， the counter freezes. When the channel is free, it will continue to counter down.
If there is a collision, double the contention window size, and wait another random time from the contention window.
Read Next packet.
```

3.2 RTS/CTS Protocol

Each station wait random time to send the packet.

When the node detect the channel is free, he will send RTS to source node and wait an SIFS. The source node will send a CTS and wait a SIFS. Then the node will send its packet. The channel is busy.

Each station can detect if the frame it sent collided with frames from other stations. If yes, choose a random time from contention window and resend again.

pseudocode:

```
Read a Packet
Wait a DIFS+ Random time from[0,15]slots
If channel is free, send the RTS to source node and wait a SIFS,  then the source will send the CTS and wait the SIFS. After receiving the RTS, the node will send its data, the channel is busy.
After sending the packet, wait DIFS and wait to get the acknowledgement, the channel is free again.
It channel is busy when the node is waiting，  the  counter freezes. When the channel is free, it will continue to counter down.
If there is a collision, double the contention window size, and wait another random time from the contention window.
Read Next packet.
```

4. Result

In this program, we suppose that the wireless system operate at a speed of 6M bps.
The number of nodes is 10; the offered load is 1.0; the average pkt_size is 100 bits; the num_pkts_per_node is 1000.
We will run 3 time for each traffic file with different methods and get their average.

A.We use exponential distribution to generate the traffic file.

a.When using 802.11 DCF MAC protocol

	1	2	3	Average
network throughput(Kbps)	938	935	934	936
total number of transmissions	8618	8664	8651	8644
total number of collisions	1382	1336	1349	1355
fraction of time the medium is free	0.430	0.431	0.432	0.431
the number of packets send for each station	1000	1000	1000	1000
average latency of packets in microseconds for each station	106	106	106	106

b.When using 802.11 RTS/CTS MAC protocol

	1	2	3	Average
network throughput(Kbps)	561	546	557	555
total number of transmissions	5216	5217	5198	5210
total number of collisions	4784	4783	4802	4790
fraction of time the medium is free	0.32	0.328	0.327	0.325
the number of packets send for each station	1000	1000	1000	1000
average latency of packets in microseconds for each station	177	179	179	178

B.We use uniform distribution to generate the traffic file.

a.When using 802.11 DCF MAC protocol

	1	2	3	Average
network throughput(Kbps)	948	955	949	951
total number of transmissions	8656	8679	8665	8667
total number of collisions	1344	1321	1335	1333
fraction of time the medium is free	0.427	0.425	0.426	0.426
the number of packets send for each station	1000	1000	1000	1000
average latency of packets in microseconds for each station	106	105	105	105

b.When using 802.11 RTS/CTS MAC protocol

	1	2	3	Average
network throughput(Kbps)	563	562	566	564
total number of transmissions	5204	5216	5175	5210
total number of collisions	4796	4784	4825	4802
fraction of time the medium is free	0.322	0.328	0.321	0.324
the number of packets send for each station	1000	1000	1000	1000
average latency of packets in microseconds for each station	177	179	178	178

5.Conclusion

Compare the result above between 2 protocols, we can draw the conclusion that

For the same traffic file at offered load 1.0, the 802.11 DCF MAC protocol will have higher throughput and lower latency than RTS/CTS.