ISSN 2250-3153

Size-Wise Back Off Period for CSMA/CD Protocol

Hitarth S Menon

B.Tech CSE Student, VIT Chennai

DOI: 10.29322/IJSRP.9.11.2019.p95103
http://dx.doi.org/10.29322/IJSRP.9.11.2019.p95103

Abstract- In the current Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol, a back off algorithm is followed upon encountering a collision between two frames sent by different systems. This algorithm involves the generating of random numbers in both the stations whose messages collided. If a re-collision occurs, the numbers are chosen from a larger range of integers. The station then has to wait for that much amount of time before retransmission.

This paper proposes a different approach to refine this process and lower the probability of collisions, by selecting the waiting period or time duration which is directly proportional to the size of the packets transmitted by the system. This will bring down the overall probability of collisions, and even avoid Ethernet Captures.

Index Terms-
Contention Window, CSMA/CD, Ethernet Capture, MFS (Maximum Frame Size), NFS (Net Frame Size), Random.

I. Introduction

The CSMA/CD algorithm follows this procedure for the transmission of messages on a channel accessed by multiple nodes. It first listens to the channel for detecting current transmissions on the line. If the channel is free, the node is allowed to transmit the message. But in the event of two systems transmitting a message at the exact same time, the algorithm detecting no current transmission on the channel will allow both systems to send the message, thus resulting in a collision.
The collision in a channel, results in a voltage displacement, which is detected by systems in the network. Upon detecting this collision, the CSMA/CD approach involves both the systems picking a random number from a range of integers (contention window) and waiting for an amount of time directly proportional to the number chosen before retransmitting their messages. If further both the systems generate the same random number, the range of the integers (contention window) is increased. Thus, the range of the contention window is directly proportional to the number of collisions occurred for a particular frame (*0 to 2 raised to the power of number of collisions occurred minus 1*).
The contention window ranges for different number of collisions is given in the following table:

Table 1

No of Collisions	Contention Window
1	0 to 2^1-1
2	0 to 2^2-1
3	0 to 2^3-1
4	0 to 2^4-1
5	0 to 2^5-1

The probability of a frame not facing collision increases after every collision.
Whatever number is chosen here (randomly), from the contention window, is multiplied by 512-bit times. Then this amount of time is spent before the node tries to retransmit the packet.

The Problem with the existing approach:

The number of collisions is limited to a maximum of 10. After which the system does not attempt to send that message again. The event of two frames choosing the exact same random numbers 10 times does seem unlikely, but the problem arises when one or more of the systems have a really large message to be sent. Consider the following case;
Two systems (say A and B) attempt sending a message at the exact same time. One of the systems (say A) has a very large size of packets. Upon collision both the systems enter into the back-off phase and wait for a randomly generated amount of time. There is only a 25% chance that B (the system with the smaller packet size) will win, i.e be allowed to transmit its packet before the system A. In case A wins, i.e starts transmitting its message, the channel will be held up for a very long period of time. This results in the channel being busy whenever the B is ready to transmit and thus the waiting time for B keeps getting longer until finally, the B loses the message it was going to transmit. This problem arises from a combination of phenomena called Ethernet Capture and Lost Node.
Thus, a system with a large amount of data will not only monopolize the channel for a long time and delay transmission of shorter frames but also may result in the complete loss of the shorter frames.

Proposed Solution:

My solution proposes two new variable nfs (net frame size) which keeps the track of the size of the frames/packets to be transmitted, and mfs (max frame size), which holds the maximum allowed size of a packet or a frame in a network.

The mfs can be set to an ideal value depending on various attributes of the network by the network administrators.
As seen in the previous sections, the usual procedure followed is: A random number is chosen from the contention window whose range keeps increasing with the increasing number of collisions faced by the particular frame. Then an amount of time directly proportional to that random number is spent before another attempt to retransmit.
But here, instead of waiting for a random amount of time, I propose that the wait time in the back-off phase is made to be directly proportional to nfs.

Formula:

Wait time = *H* + { (*nfs*/*mfs*) x 512-time units x *M* }

Nfs: Net frame size
Mfs: Maximum frame size
H, M: constants

The values for mfs and the constants M and H are determined beforehand. These values can be kept constant or can be modified according to the necessity.
The value of mfs can be increased if the network has to carry larger frames. If the network is expected to carry short messages but with a fast speed then the value of mfs can be made smaller.
The constant value of M can be increased if there are many nodes trying to send a frame and a longer range of wait time is required. The value of M can be made smaller if the wait time has to be reduced and a faster network is needed.
The constant value of H, is the minimum wait time, for which the system has to wait. This can either be decided by network authorities or the system itself. By default, H is to be 0 for all systems on the network.
The value of nfs is variable and can be only determined by the system transmitting the frame.

Advantages:

Thus the waiting time of the back off phase instead of being a randomly generated time unit, is a value that is directly proportional to the frame size to be transmitted.
This new formula would enhance the current CSMA/CD protocol in the following ways:

1. Lowering the probability of collisions: Every frame being transmitted will have different size. The sizes may not always be unique, but the probability of two frames which transmit at the exact same time and also have the exact same size is fairly low. This will bring down the number of re-collisions by a large extent.
2. Avoiding Ethernet Capture and Loss of Frames: Since the waiting time is to be directly proportional to nfs (the message size), this algorithm makes sure that in most cases, the system with the smaller packet/frame size, gets a chance to utilize the channel. Since a smaller message will take less time for transmission and propagation, it will release the channel faster and let other systems transmit their messages. This ensures that an Ethernet Capture doesn't occur in the channel.
3. Lower Waiting time: The range of the contention window in the original CSMA/CD protocol was 0 to 2^n with n being the number of collisions. With this algorithm the range of values can be made much smaller. That is done by making sure that the M value is set to less than 2^n.
4. Completely customizable range of wait time: The constant values of H and M can be altered and the duration of the back off period can easily be changed to any value.
5. Prioritizing important messages: The network administrator decides which messages are potentially important and can alter the H and M values of that node to a minimum to ensure that potentially important messages don't have to wait for a longer period of time. None of the previous algorithms allow the prioritizing messages.

All these factors will play a major role in the smooth functioning of a channel. This algorithm however has some disadvantages that need to be worked upon.

Disadvantages:

1. If 2 frames have the same size, some anomaly may be caused in their transmissions. Even though the probability of 2 frames with the exact same size trying to transmit at the same time is fairly low, such an even is still possible.
2. A message that has a large frame size will have to wait for a longer time if a collision occurs. If that frame happens to carry any crucial information, then it may be delayed.
3. The above two shortcomings can be completely overcome by varying the constants in the formula. But it requires continuous surveillance by the network administrators.
4. In the formula given above, anomalies would be caused if the size of a particular frame (nfs) happens to be larger than the estimated maximum frame size (mfs).

II. CONCLUSION

From the given disadvantages, the first two can be easily dealt with by the network administrator varying the constants H and M in the formula. But this would require the network administrators to continually monitor the entire process.
But with the continuous advancements in the fields of Machine Learning and Prediction Models, these drawbacks too can be replaced by machine learning algorithms that learn to handle the constants in the systems, and set them to an ideal value, and ensure faster functioning of the network.

ACKNOWLEDGMENT

A special thanks to the faculties of VIT Chennai for their undying encouragement to innovate. A special mention to Prof. Dr. Reshmi TR (VIT Chennai School of Computing, B.tech,(IT) ME (CSE)) for kindly reviewing this paper.

AUTHORS

First Author – Hitarth S Menon, pursuing B.Tech CSE, VIT Chennai, hitarths.menon2018@vitstudent.ac.in