CSMA/CD Variant with Successive Collision Probability Multiplication

Yash Dekate

B.Tech Computer Science, Vellore Institute of Technology, Chennai

DOI: 10.29322/IJSRP.9.11.2019.p9528
http://dx.doi.org/10.29322/IJSRP.9.11.2019.p9528

Abstract- CSMA/CD is basically a collision dealing arrangement which comes with a set of predefined persistent methods. It detects these collisions by sensing the medium first before sending the data and if there is collision, it immediately aborts the transmission and then later tries to send the data again according to any of the predefined persistent methods. Though this ensures the proper and safe transfer of data but when there are multiple nodes which have some data to send, it can happen that nodes might have to wait a longer time to transmit their data. Hence, decreasing the efficiency of the overall system. Here I'll be putting forward my approach for dealing with this problem with a tweak of back-off algorithm which overcomes its basic shortcomings and some more general properties of CSMA to design a more efficient system than CSMA/CD.

I. Introduction

CSMA/CD basically deals with the process of transferring data between nodes by sensing the medium before transferring the data. If it receives a signal that the medium is ideal, it sends its data immediately. But, at this time is also sometime happens that the other node also senses the channel ideal and send its data. Hence, this results into a collision and it is detected by the arrangement and the current data transfer is aborted. Now, the data is tried to be resent according to 3 persistent methods

- 1-Persistent Method

The system continuously checks for the medium to be ideal and if it is, it sends the packet with probability of 1.

- Non-Persistent Method

The system checks system after random amounts of time. If the channel is found free, it sends the data immediately.

- p-Persistent Method

The system senses the medium and if found ideal, sends the data with a probability of p and if there is a collision (1-p), it retries after a certain amount of time again with probability p. This goes on till the data is sent properly.

Now, though this system is great, it is inefficient in cases of huge traffic and for this many variants of the CSMA/CD are being proposed to overcome the traffic problems and extra bandwidth usage. Here, I'll be describing my proposed variant for the same.

II. Problem Statement

Though CSMA/CD detects the collisions and deals with it, it is very inefficient when there are many collisions happening in the medium, i.e., the traffic is high. Hence, what happens is there are a lot of collision detections and hence, a lot of time elapses before the required node transmits its data completely and safely. This is because same node can suffer multiple collisions for its data and hence it may take a long while to transmit the data properly to the target node.

III. Related Research

A number of CSMA/CD variants exist but the one I've researched for my proposed variant is the, 'Back-Off Algorithm' for CSMA/CD

1. Back-Off Algorithm

The back-off algorithm basically just associates the nodes with numbers after every collision and then detects the probability of collision on the basis on the assumed inputs. After each successful transmission of data, the probability of collision goes on decreasing which is quite desirable. But, this has two major drawbacks

- If one node wins and transfers its data, then again it is possible that the same node can transfer data. This may result into a collision again.
- This method is only limited to 2 nodes.

I'll be putting forth my variant to basically overcome these drawbacks by some or the other ways.

IV. Solutions & Analysis

If we notice closely, all the penalties of the time elapsed, extra bandwidth usage, etc. is actually caused by the greater probability of collision and the successive collisions between the same nodes. Hence, our goal should be to basically reduce the probability of the collisions with each pass and introduce a token system transmit the data with the proper tokens. Moreover, to ensure maximum transmission of data per pass, we could transfer multiple frames of data together at once by keeping its maximum size as 1500 bytes as this is the defined limit.

To reduce the probability of collisions, we could use some math. We could look to use a function which decreases the probability of collision for each pass of data and at the same time use a token system to ensure that no successive collisions occur and sending as much data as possible in sets of 1500 bytes. Hence, my solution exploits 3 things mainly

- Decreasing P(Collisions/Pass)
- Involving a token system to prevent successive collisions between the same nodes
- Transmitting data in maximum packet sizes

I'll be discussing how we'll accomplish these

- Decreasing Probability of collisions per pass & involving a token system to prevent successive collisions between the same nodes.

1. FIRST PASS

To decrease the probability of collisions, we will associate each node with {0,1}. Note that here, we can associate only two nodes with them. Hence, when there are more than 2 nodes, we can divide it into sets of 2 and then proceed for the individual sets separately.

Now, let there be one collision and after than let the nodes A & B assume 0s and 1s as

A	B
0	0
0	1
1	0
1	1

Here, probability of collision is ½. (When A & B are equal)

Now, let A win. Then it will transfer its data and here a token which is already held by A is passed to B so that now compulsorily B has to transfer its data. This avoids the problem that there may be a subsequent collision after A wins in the previous step and in case A tries to send a packet again.

Now, let B be associated with {0,1} and C be associated with {0,1,2}

Hence, the probability of collision now becomes 1/3. Here, B wins, transfers its data and then passes the token to C.

Now, let C be associated with {0,1,2} and D be associated with {0,1,2,3}.

Hence, the probability now become ¼. This goes on till it reaches its final target node.

Hence, net probability is

P(First Pass Collisions in First Pass) = 1/2 x 1/3 x 1/4 x … (Till the last node)

Now, for the second pass, A is associated with the probability of 1/3, B with 1/4 and so on. Hence. Net probability in this case is

P(Second Pass Collisions in First Pass) = 1/3 x 1/4 x 1/5 x … (Till the last node)

Here we notice that with each transfer of a packet from A to any other node, its probability of collision is decreasing. This aids us in reducing the collision frequency and hence, faster transfer of data.

2. SECOND PASS

Now, for the second pass, the probability of A transmitting the data is already set to 1/3 as the transmitting starts with A being associated with {0,1} and B being associated with {0,1,2} and it goes on in a similar way and the probability of collision goes on decreasing.

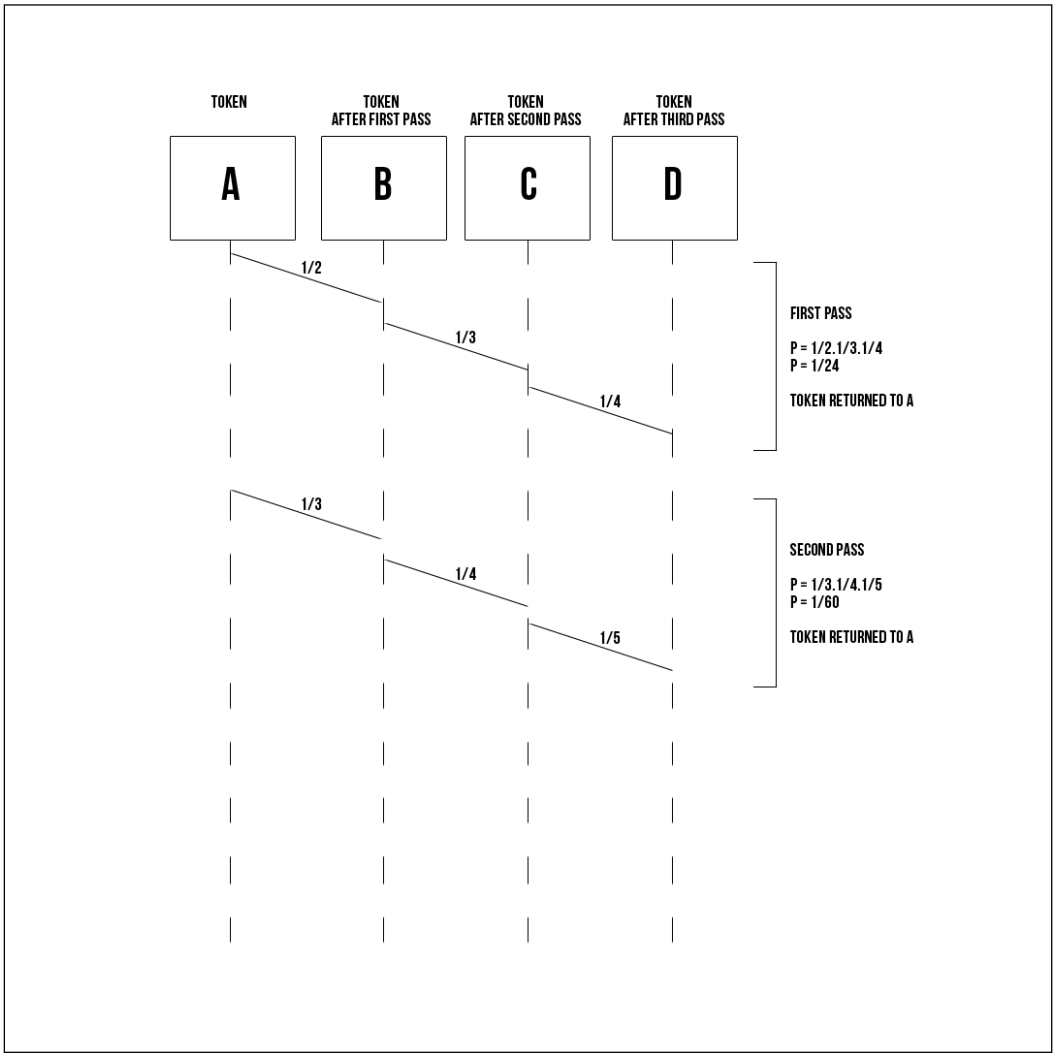
Probability of each node starts from the calculated value of probability and the subsequent probabilities are calculated as specified by the process above.

Hence, for each pass, the probability of collision goes on decreasing and hence, the data is transmitted faster.

- Sending data in maximum packet sizes

I'm exploiting one more point here. The point if that the maximum size of the data that can be transmitted is 1500 bytes. Hence, we can prepare sets of frames which have a net size of <= 1500 bytes and we will transfer these through the channels and not separate frames one by one.

The diagrammatic representation of the process is as follows



For better clarity of the picture, please visit this link.
https://ibb.co/q02L2jG

Comparing with back-off algorithm, my variant, overcomes those by doing the following

- Due to successive introduction of state numbers or alphabets and the concept of probability, the collision probability becomes lesser and lesser in each case.
- Due to introduction of the token system, the possibility of A retransmitting the data after a successful transmission is ruled out as the token is then passed to B. Hence, this reduces the unnecessary collision which may take place.

However, there are also some disadvantages of the proposed variant

- If any of the nodes fail, the whole system may stop working.
- We can't keep on adding the numbers associated with each node as for everything, there is a limit.

V. Summary

Though CSMA/CD is quite useful, it has flaws which my proposed variant is able to resolve to some extent. My proposed variant may prove quite useful if implemented properly.

References

[1] Data Communications and Networking By Behrouz A Forouzan
[2] CSMA
https://www.geeksforgeeks.org/carrier-sense-multiple-access-csma/
[3] Multiple Access Protocols in Computer Networks
https://www.geeksforgeeks.org/multiple-access-protocols-in-computer-network/
[4] Back-Off Algorithm for CSMA/CD
https://www.geeksforgeeks.org/back-off-algorithm-csmacd/

Authors

First Author – Yash Dekate, B.Tech Computer Science, Vellore Institute of Technology, Chennai