Enhanced CSMA/CA Using Logarithmic Increase Multiplicative Decrease Backoff Strategy

Atif M. Saleed, Sanju George Oommen

Abstract

Over the last two decades, the internet has become a daily necessity for billions of people. Of these users, wireless mobile phones and laptops take up the majority of devices. Wireless networking has become increasingly popular in the span of the past two networks and the need for reform is ever-present. Pre-existing protocols like CSMA/CA which are particularly useful for wireless networks have been in use without contention for a long time. CSMA/CA looks around in the network for data to be sent and sends only when the traffic in the network will not be obstructive. In case the condition isn't met, the device waits for a randomly generated backoff time within a defined range called a contention window, which changes based on previous history. In this paper, we explore the various strategies used for backoff time and suggest an improvement to a pre-existing backoff strategy.

Introduction

CSMA stands for carrier-sense multiple access and is a protocol used to avoid collision, detect collision, or send notification for collision in the network medium. CSMA/CA is one such protocol that tries to avoid collision by checking for traffic in the network using RTS and CTS signals before sending data. Once no traffic is observed, the packet is ready to be sent and the system waits for a random value of time in the range [CW_{min}, CW_{max}] before finally sending it.

However, when traffic is observed certain modifications are made to the aforementioned contention window. The contention window size is typically modified in two cases:

1. Successful receival of the CTS signal or transmission of the packet.
2. Unsuccessful attempt to attain CTS and the transmission is set to wait.

Modifications are made to the contention size window in both the above cases. In is decreased in the first case (when traffic isn't observed) and increased in the second case (when traffic is observed). We explore network utilization for five different backoff strategies:

1. 802.11 Standard
2. Additive Increase Additive Decrease
3. Additive Increase Multiplicative Decrease
4. Multiplicative Increase Additive Decrease
5. Multiplicative Increase Multiplicative Decrease

The 3rd strategy (Multiplicative Increase Additive Decrease) can be described by the below formula:

$w(t+1) = w(t) + a$ if congestion is not detected
$w(t+1) = w(t) * b$ if congestion is detected

where $w(t)$ is the sending rate

These days, typically the additive increase parameter (a) is typically the time taken for 1 MSS (maximum segment size).

Proposed Algorithm

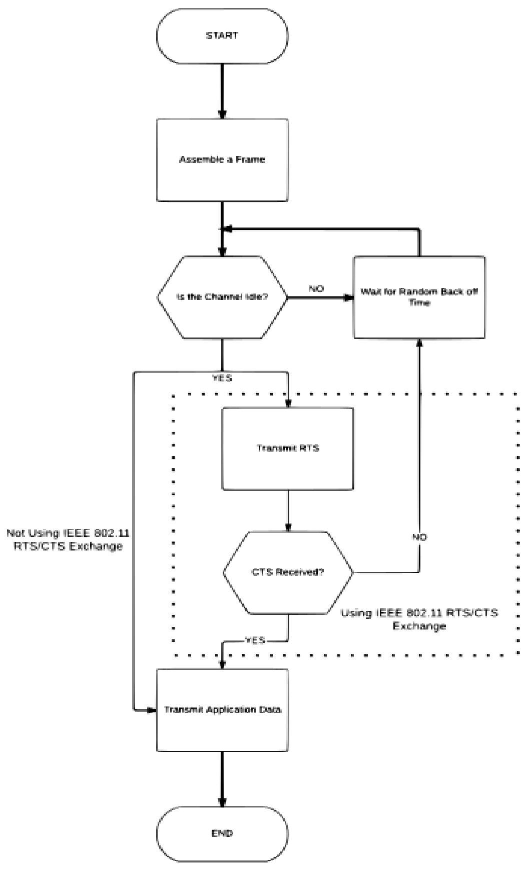
$w(t+1) = w(t) + \max(a, k * \log(w(t)))$ if congestion is not detected
$w(t+1) = w(t) * b$ if congestion is detected

In our proposed algorithm, the increase in sending rate is a function of the previous sending rate.

Consider the case of AIMD where the sending rate is increased by the same unit, regardless of the previous sending rate. This unit increase in sending rate is often inconsequential for large values of sending rate. By using the log of sending rate, the increase in sending rate is dependent on sending rate, and of more impact.

We simulate and extract metrics for all backoff algorithms, using the proposed algorithm in place of the usual AIMD.

Proposed Work


START
Assemble a Frame
Is the Channel Idle?
NO
Wait for Random Back off Time
YES
Transmit RTS
Not Using IEEE 802.11 RTS/CTS Exchange
CTS Received?
NO
Using IEEE 802.11 RTS/CTS Exchange
YES
Transmit Application Data
END

CSMA/CA Algorithm

Proposed Algorithm

$w(t+1) = w(t) + \max(a, k * \log(w(t)))$ if congestion is not detected
$w(t+1) = w(t) * b$ if congestion is detected

In our proposed algorithm, the increase in sending rate is a function of the previous sending rate.

Consider the case of AIMD where the sending rate is increased by the same unit, regardless of the previous sending rate. This unit increase in sending rate is often inconsequential for large values of sending rate. By using the log of sending rate, the increase in sending rate is dependent on sending rate, and of more impact.

We simulate and extract metrics for all backoff algorithms, using the proposed algorithm in place of the usual AIMD.

Pseudo Code

```
current_time = 0
while (current_time < sim_time):
    assign each node a slot randomly, within contention window
    m = min(slot1, slot2 .... slotn)
    if m overlaps with another slot
        collision_flag = 1
    else
        collision_flag = 0
    if collision_flag = 1:
        Update contention window
        Reassign slots to colliding nodes
    else:
        Increment good_time by time ticket for packet
        Clear packet for previously found min. node
        Increment current_time


network_utilization = good_time/current_time
```

The proposed algorithm works similar to the commonly used additive increase multiplicative decrease strategy for transmission speed, the only different being that logarithmic increase is used instead.

Logarithmic increase works better than additive increase due to face that logarithmic decrease pays attention to the previous size of the contention size as opposed to additive increase that increases transmission time by the same amount regardless of contention size. With logarithmic increase, there is a lesser chance of the constant additive increase being inconsequential to pre-existing contention window size.

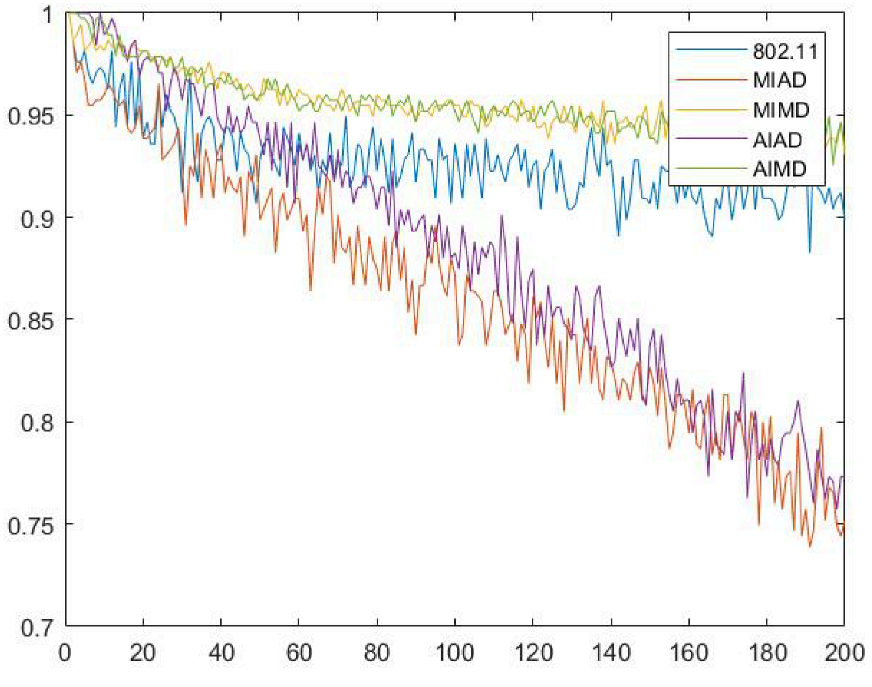
Simulation Results & Analysis

For the simulation, we make the assumption that slotted CSMA and real-time CSMA produce the same results as far as backoff strategy is concerned.

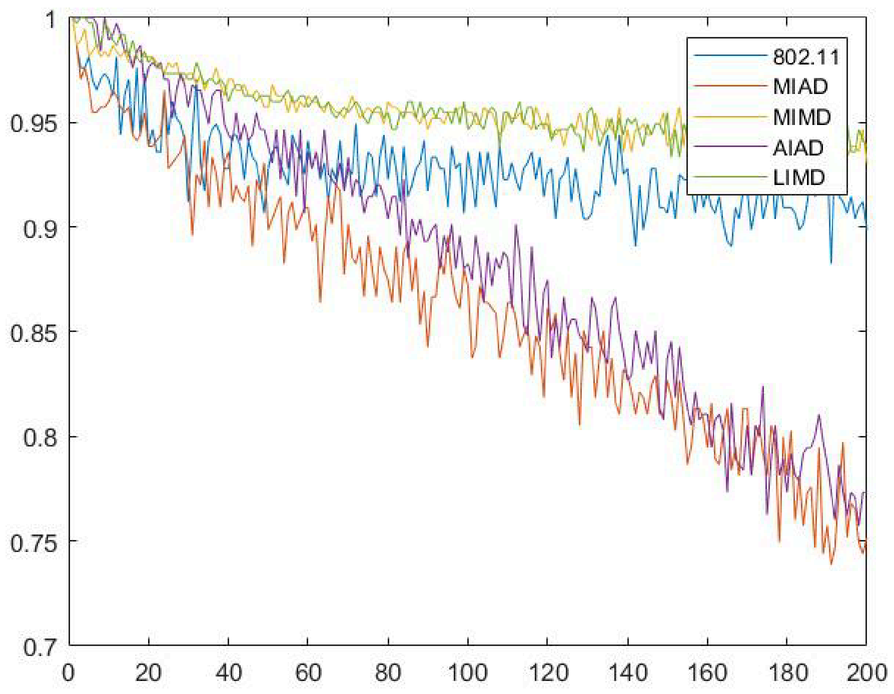
The backoff implementation assigns each node a particular slot and clears each node's packet, starting from the node with the earliest slot assigned provided there is no collision with other nodes. In this case, the time taken for packet transmission is added to utilized time, which will be used to calculate network utilization for a given backoff strategy and a given number of nodes.

In the case of overlap in slots for the earliest occurring slot, we change contention window and assign these nodes a different slot accordingly.

Packet size	2000 bits
Simulation time	1000 seconds
Number of nodes	1 – 200
Slot size	9 microseconds
Data rate	6 Mbps
Initial contention window size	15



Network utilization with varying number of nodes using pre-existing backoff strategies



Network utilization with varying number of nodes using LIMD in place of AIMD

Since AIMD already offers the best network utilization among the different strategies, improvement using LIMD isn't very evident from the above graph.

We run both the original and improved algorithm 1000 times and score both strategies looking at how often each strategy offers better network utilization. Such a setup is appropriate since comparing difference network utilization alone given a number of nodes is subject to large degrees of variance, due to randomly generated slots and the unpredictable working of such strategies.

Due to the mentioned randomness, it is hard to effectively quantify minute differences between both strategies.

Strategy	LIMD	AIMD
Points given 100 nodes	518	481
Points given 200 nodes	509	491

Conclusion

CSMA has clearly demonstrated being one of the most prevalent standards in use for wireless networks in this age. The most common of currently adopted backoff strategies, AIMD (Additive Increase Multiplicative Decrease) has proven to be one of the most reliable as far as network utilization and congestion control is concerned. Here, we demonstrate LIMD(Logarithmic Increase Multiplicative Decrease) as a possible improvement to AIMD and as an additional consideration to improve throughput. As mentioned earlier, the proposed algorithm should be tested over larger simulations, since minute improvements are harder to quantify with readily available hardware and simulation software. A MATLAB simulation is used to demonstrate and compare network utilization of different strategies and improvement over using the proposed method.

References

LIAD: Adaptive bandwidth prediction based Logarithmic Increase Adaptive Decrease for TCP congestion control in heterogeneous wireless networks
https://www.sciencedirect.com/science/article/pii/S1389128609001704

Logarithmic Based Backoff Algorithm for MAC Protocol in MANETs
https://pdfs.semanticscholar.org/b84e/b1b2c5ac92c146f0c4d3f9a93c518d80a82b.pdf

Logarithmic window increase for TCP Westwood+ for improvement in high speed, long distance networks
https://www.researchgate.net/publication/222541994_Logarithmic_window_increase_for_TCP_Westwood_for_improvement_in_high_speed_long_distance_networks

CSMA with Enhanced Collision Avoidance: A Performance Assessment
https://www.researchgate.net/publication/224504774_CSMA_with_Enhanced_Collision_Avoidance_A_Performance_Assessment

Performance analysis of IEEE 802.11 CSMA/CA medium access control protocol
https://ieeexplore.ieee.org/abstract/document/567426

Enhanced backoff scheme in CSMA/CA for IEEE 802.11
https://www.spiedigitallibrary.org/conference-proceedings-of-spie/5100/0000/Enhanced-backoff-scheme-in-CSMACA-for-IEEE-80211/10.1117/12.486778.short?SSO=1

On Optimization of CSMA/CA based Wireless LANs: Part I - Impact of Exponential Backoff
https://ieeexplore.ieee.org/abstract/document/4024473

Analysis of the "Additive Increase Multiplicative Decrease" Model for Congestion Avoidance
https://pdfs.semanticscholar.org/0266/ac8fccec557e7d65ba9152dbf7d174ef46c4.pdf

Improving the Collision Avoidance of the CSMA/CA Medium Access Control Protocol
https://www.researchgate.net/publication/242638743_Improving_the_Collision_Avoidance_of_the_CSMACA_Medium_Access_Control_Protocol

CSMA with Enhanced Collision Avoidance: A Performance Assessment
https://ieeexplore.ieee.org/document/5073486

An analysis of the backoff mechanism used in IEEE 802.11 networks
https://ieeexplore.ieee.org/abstract/document/860678