WLANs throughput improvement with CSMA/ECA

Luis Sanabria-Russo, Jaume Barcelo, Azadeh Faridi, Boris Bellalta
Department of Information and Communications Technologies
Universitat Pompeu Fabra, Barcelona, Spain
{luis.sanabria, jaume.barcelo, azadeh.faridi, boris.bellalta}@upf.edu

***Abstract*—Carrier Sense Multiple Access with Enhanced Collision Avoidance (CSMA/ECA) is a distributed MAC protocol for WLANs, capable of achieving greater throughput than the current contention mechanism in WLANs. It does so by changing to a deterministic backoff after successful transmissions, which leads to a collision-free schedule that under ideal conditions can be permanently maintained. This demo shows the first implementation of CSMA/ECA using commercial hardware and OpenFWWF in a realistic network testbed. Results show how CSMA/ECA outperforms the current MAC for WLANs in terms of throughput, even through a permanent collision-free schedule cannot be maintained due to unideal practical conditions.**

***Index Terms*—CSMA/ECA, WLAN, MAC, Collision-free, OpenFWWF.**

I. Introduction

In this demo we show the operation of a WLAN using Carrier Sense Multiple Access with Enhanced Collision Avoidance (CSMA/ECA) [1] and how it outperforms the same network setup using the current MAC for WLANs. The testbed consists of four stations transmitting messages to a wired Server/Sniffer station through a commercial Access Point (AP) (see Figure 1).

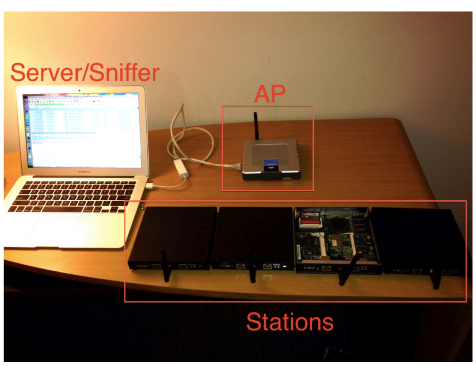

Fig. 1. Demo setup: four Alix 2d2, Iperf server and an AP.

During the demo, the stations are running CSMA/ECA as their MAC protocol. The Sniffer records the transmissions' timestamps, transmitter identity, and the number of received packets. This information is then processed at the end of each test to provide metrics such as: throughput, retransmissions, successful transmissions, and average frame inter-arrival times for each station.

II. Carrier Sense Multiple Access with Enhanced Collision Avoindace

CSMA/ECA behaves exactly as Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), except that after a successful transmission the transmitting station picks a deterministic backoff, B_d, and transmits periodically until a failed transmission. A failed transmission is indicated by the absence of an acknowledgement (ACK) from the receiver and may be caused by collisions or channel errors.

As in CSMA/CA, a node involved in a failed transmission increments its backoff stage, k, by one ($k \in [0, m]$, where m is the maximum backoff stage of typical value $m = 5$) and picks a random backoff, $B \sim \mathcal{U}[0, 2^k CW_{\min}]$, where $CW_{\min}$ is the minimum contention window with typical value $CW_{\min} = 16$.

Figure 2 shows the CSMA/ECA behavior under ideal conditions. Four stations (STA) are involved in a contention to access the channel using CSMA/ECA. The horizontal lines represent time and are composed of empty slots and transmissions. Each empty slot decrements the backoff by one, so the numbers indicate how many empty slots are left until the expiration of the corresponding STA's backoff. At the first slot, the outline points out that STA-3 and STA-4 have picked the same random backoff and will eventually collide. Upon collision, these two stations will recompute a random backoff.

It is not until a station is able to make a successful transmission that it changes to a deterministic backoff. In Figure 2, STA-4 is able to successfully transmit after the random backoff expires, and then it generates a deterministic backoff ($B_d = 7$) for the next transmission. Under ideal conditions, transmissions fail only due to collisions. In this case, once all stations transition into the deterministic backoff mode, a perpetual collision-free schedule is reached. Further extensions to CSMA/ECA enable it to construct a collision-free schedule for many more contenders [2].

III. Demo

We prototyped a CSMA/ECA station using OpenFWWF firmware into Broadcom BCM4318 chipset Wireless Network Interface Controller (WNIC), which is connected to a mini-PCI slot inside a PC Engines Alix 2d2 [3] station. By making simple changes to the OpenFWWF [4] open firmware for WLAN network cards, the built-in MAC can be modified to mimic CSMA/ECA behavior. This CSMA/ECA implementation in commercial hardware will be referred to as CSMA/ECA$_{\text{test}}$ hereafter. Further implementation details using commercial PCs can be found in [5].

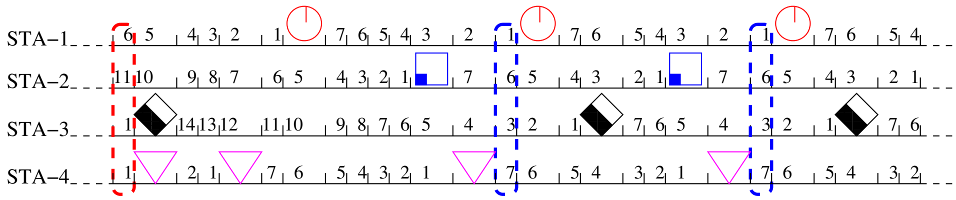

Fig. 2. CSMA/ECA with four stations in saturation ($B_d = 7$).

The testing scenario is composed of four saturated stations running CSMA/ECA$_{\text{test}}$. Each station is placed at similar distance from the Access Point (AP), to which an Iperf [6] server is connected via Ethernet, as in Figure 1. Stations are set to transmit dummy 1470 byte UDP segments at 65 Mbps towards the server, and the transmissions are captured using Wireshark [7] in a separate wireless station so they can be visualized in real time or saved for processing at a later time. The transmission speed is set to 65 Mbps to purposely saturate the stations.

In order to extract useful statistics from the capture files, they are exported to Comma Separated Values (CSV) format files and processed using a parser written in Python, which is available online [8].

IV. Results

Figure 3 shows a comparison between CSMA/ECA$_{\text{test}}$ and CSMA/CA implemented in the same testbed. Even though CSMA/ECA$_{\text{test}}$ cannot maintain a collision-free schedule due to unideal practical conditions, it can be seen that it achieves better throughput than CSMA/CA, due to a reduction in the number of collisions. This is because stations that have switched to a deterministic backoff enjoy a lower probability of collision. The dashed line represents the achievable throughput of a CSMA/ECA station under ideal conditions (no channel errors or clock drift).

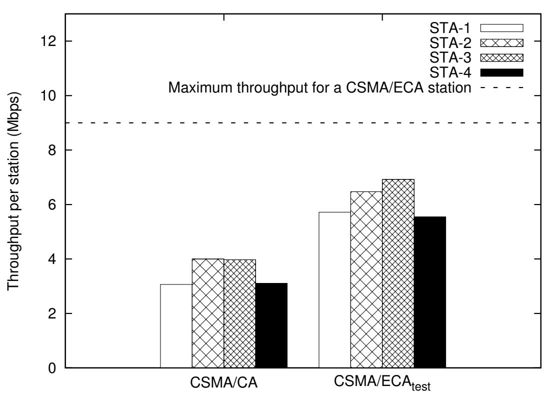

Fig. 3. CSMA/ECA$_{\text{test}}$'s approximately 50% improvement over CSMA/CA stations. Both protocols have $CW_{\min} = 32$ and for CSMA/ECA$_{\text{test}}$, $B_d = 16$. This guarantees a fair comparison by ensuring the same expected backoff value after successful transmissions in both protocols.

Under unideal conditions, a collision-free schedule is occasionally reached but it is disrupted as soon as a transmission is corrupted by noise, or when there is a slot drift due to hardware clock inaccuracies. Figure 4 shows a sequence of transmissions over a ten-millisecond period. In this particular snapshot, all stations are transmitting with deterministic backoff and a periodic schedule is observed. However, a drift can be appreciated in STA-1's second transmission, which in time can cause a collision and disrupt the collision-free behavior.

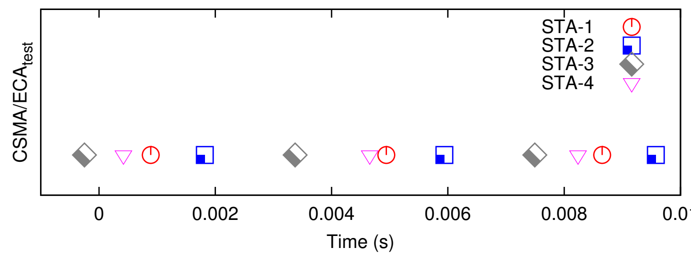

Fig. 4. CSMA/ECA$_{\text{test}}$ transmissions under saturation. In this figure $B_d = 256$, for increased robustness against slot drift.

V. Conclusions

Channel errors, clock drift, and other hardware imperfections prevent CSMA/ECA from reaching a collision-free schedule. However, this demo shows that in a practical setup CSMA/ECA still outperforms CSMA/CA. The performance improvement is due to the fact that deterministic stations do not contend against each other and enjoy a reduced collision probability.

Further performance improvements are possible by prototyping the same protocol in hardware with more accurate clocks.

References

[1] J. Barcelo, B. Bellalta, C. Cano, and M. Oliver, "Learning-BEB: Avoiding Collisions in WLAN," in *Eunice*, 2008.

[2] L. Sanabria-Russo, A. Faridi, B. Bellalta, J. Barceló, and M. Oliver, "Future Evolution of CSMA Protocols for the IEEE 802.11 Standard," in *2nd IEEE Workshop on Telecommunications Standards: From Research to Standards. Budapest, Hungary*, 2013.

[3] PC Engines. (2013) PC Engine Alix 2d2. System board. Webpage. [Online]. Available: http://www.pcengines.ch/alix2d2.htm

[4] F. Gringoli and L. Nava. Open Firmware for WiFi Networks. Webpage. [Online]. Available: http://www.ing.unibs.it/openfwwf/

[5] Sanabria-Russo, L. (2013) Report: Prototyping Collision-Free MAC Protocols in Real Hardware. Webpage. [Online]. Available: http://luissanabria.me/written/beca-test1.pdf

[6] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "Iperf: The TCP/UDP bandwidth measurement tool," *http://dast.nlanr.net/Projects*, 2005.

[7] Combs, Gerald and others. (2007) Wireshark. [Online]. Available: http://www.wireshark.org

[8] L. Sanabria-Russo. (2013) Parsing CSV files from Wireshark. [Online]. Available: https://github.com/SanabriaRusso/sharktoolStatistics