

# Simulation Analysis about Higher Performance of GDCF Over DCF in IEEE 802.11 MAC Layer Protocol

Feng An and Dongsoo Kim

*Abstract*—Wireless communication is being developed in the last years day by day, there are several standards that talks about it. We are going to go through the IEEE 802.11 which talks about wireless LAN Medium Access Control (MAC) Specifications,

This paper introduced an enhanced version of DCF, called gentle DCF. In the ordinary IEEE802.11 DCF protocol, the IEEE802.11 DCF decreases the contention window to the initial value after each success transmission, which essentially assumes that each successful transmission is an indication that the system is under low traffic loading. GDCF takes a more conservative measure by halving the contention window size after  $c$  conservative successful transmissions. This gentle decrease can reduce the collision probability. The GDCF compute the optimal value for  $c$ , and the numerical results from both analysis and simulation demonstrate that GDCF significantly improve the performance of 802.11 DCF.

The work that we are trying to do is to verify GDCF using our own wireless simulation model, we modify the original DCF using C++ language and testing it with NS (network simulator). This test program is written in TCL script. We will simulate a wireless network where we are going to have a short number of stations in a given field and try to analysis the revised DCF by comparing the performance of the revised one with that of the original DCF.

I finish the design and implementation of GDCF (gentle DCF) on Quarry, Indiana University's super computer, performance results are given in the subject of mean throughput, mean delay, delay statistics, delay dependence on distance. It also performs a graph analysis for some important parts of the implementation of GDCF.

*Index Terms*— C/C++ DCF GDCF IEEE 802.11 NS2 Performance Evaluation Perl

## I. INTRODUCTION AND RELATER WORKS

This paper focuses on the contention-based MAC protocols used in wireless local area network, specifically IEEE 802.11 DCF [1, 2, 6]. The analysis in [3, 4, 5] demonstrated that the throughput and fairness of 802.11 DCF could significantly deteriorate when the number of nodes increases, while GDCF can successfully deal with this problem, and at the same time lower the packet drop rate.

## II. PERFORMANCE EVALUATION OF GDCF WITH ITS COMPARISON TO THAT OF DCF

I get files from Prof. Dongsoo Kim, and modify the NS 2.30 on my own computer and also NS-2.33 on the Quarry super computer. I then write a test program to test my GDCF design and demonstrate the superiority of GDCF over DCF.

The test program meet those criteria:

The size of the network geometry is  $1000m \times 1000m$ ,  $N$  mobile nodes are randomly placed in the region, and they can move around the region based on either RW or RWP. Each pair of nodes, with a probability  $p$ , continuously send and receive packets at the rate  $R$  kbps, whose packet size is fixed, say  $L$  bytes, using one of the routing protocol available from NS. Each node must have at least one sender and one receiver.

I need to conduct two experiments:

Experiment 1: network performance depending on the number of nodes in DCF and GDCF

Experiment 2: network performance depending on the packet size in DCF and GDCF.

I have set up five components for each experiment to analysis the GDCF and DCF in NS2.

**Experiment 1:** Network performance depending on the number of nodes in DCF and GDCF:

I change the node number from 2 to 13, and try to run the test script in DCF and GDCF (GDCF4 and GDCF8), in

this experiment, I set the packet size to 500 bytes, cbr rate into 1mb.

### Mean throughput:

I use perl to analysis the throughput of the tracefile, I set the calculation precision into 1 unit time, at the end of each unit time, for example, simulation time point 1, 2, 3, ...etc. The perl file will calculate the number of packets generated and the size of each packet and then calculate the throughput at that simulation point, then at the end of the simulation, the perl file will print out the mean throughput of the whole simulation.

Here are the results:

Node number	DCF (bps)	GDCF4 (bps)	GDCF8 (bps)
2	17211.68	17171.28	17128.29
3	56643.88	55993.77	56004.13
4	148074.42	153657.8	153174.91
5	387508.9	416319.54	413963.96
6	693275.94	694095.84	693760.2
7	981113.85	982982.55	979396.84
8	1486285.36	1536699.12	1539988.81
9	2352216.56	2363173.05	2279567.76
10	2722939.55	2644209.44	2711562.76
11	4325115.8	4462971.89	4372532.27
12	6021927.05	5973282.57	5963043.71
13	7010535.39	7029984.82	7118902.1

According from the chart, we can see that the mean throughput of GDCF and DCF are approximately the same. It may due to a lack of mobile nodes, because in the GDCF documentation, it says that the GDCF will usually increase the throughput.

### Mean Delay:

I use tracegraph, an integrated calculation software to calculate the mean delay of each simulation.

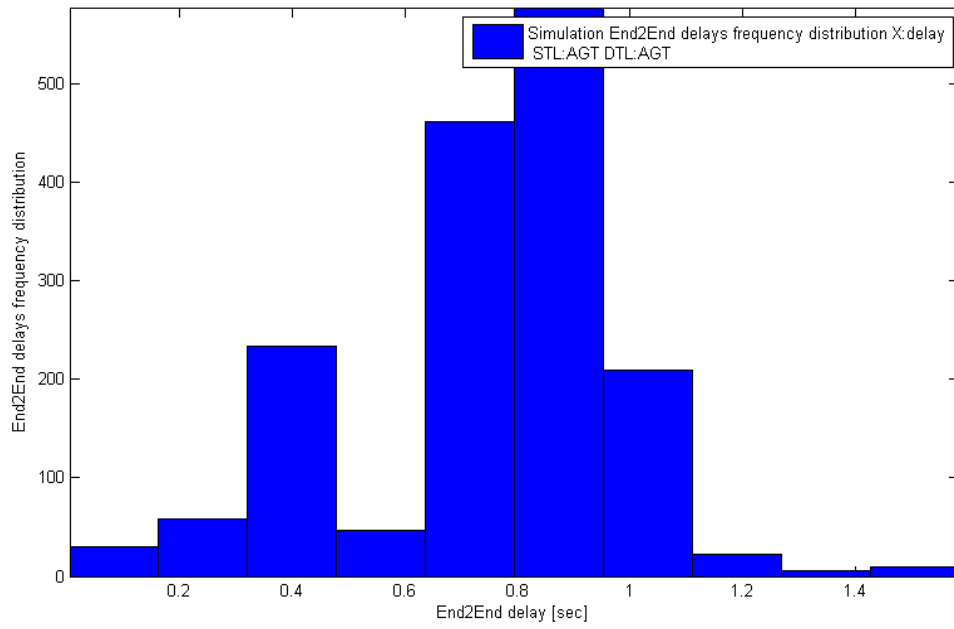
Here are the results:

Node number	DCF	GDCF4	GDCF8
2	0.5260844774	0.5272965789	0.5289298907
3	0.7474287774	0.7140991742	0.6304053008
4	0.9417049018	0.6194140683	0.8796291255
5	1.10572751	0.547411207	0.5257803473
6	1.277226585	0.8668664771	0.7938767828
7	1.481345942	1.16293758	1.190302433
8	1.450570798	0.9232749365	0.8736601791
9	1.913381191	0.9850461888	1.016257738
10	1.079364501	0.7899646881	0.8703672065
11	1.31345593	0.9676323358	0.8904535144
12	1.328172823	1.043857915	0.9370507995
13	1.125375582	0.6030551918	0.8799317526

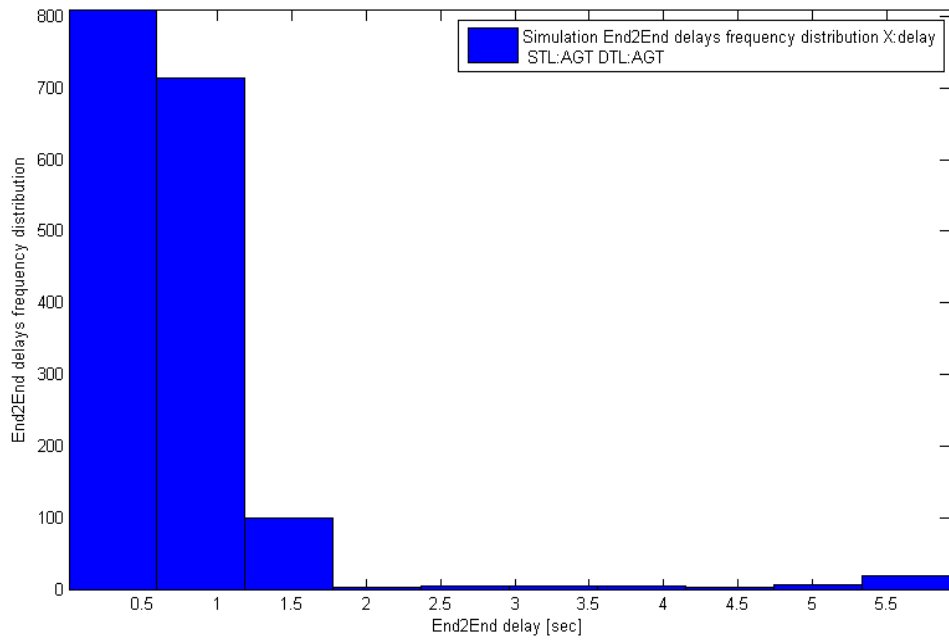
As shown in the statistics window, the DCF's value significantly increase when the node number increase, while the mean delay in GDCF did not increase as much as in the DCF. The GDCF documentation results have been tested out.

### Delay statics:

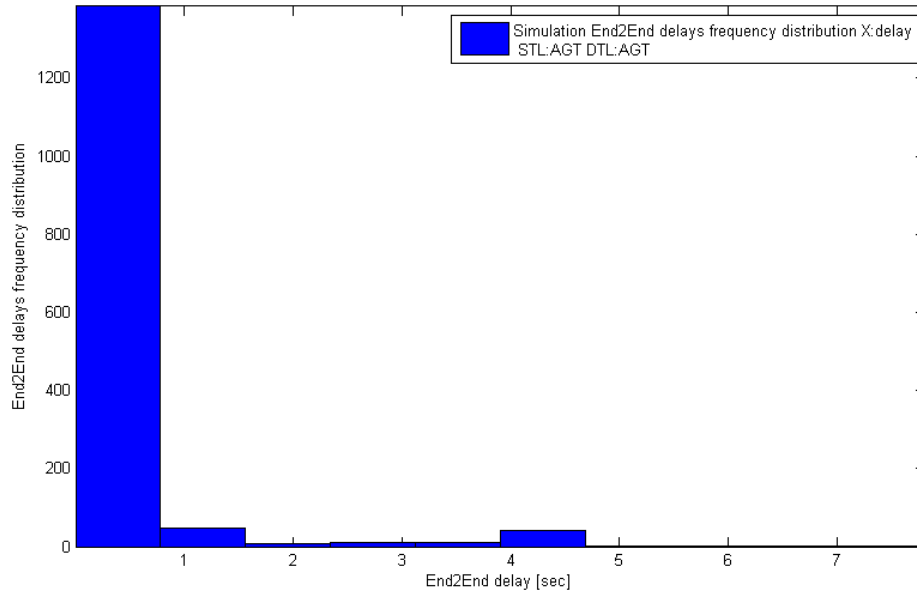
I use tracegraph to finish the delay statistics calculation. As the same trend has appear in each experiment, which different node numbers are applied. I choose 3 as an example to show the difference between DCF, GDCF4 and GDCF8. Please see the chart below:



DCF



GDCF 4

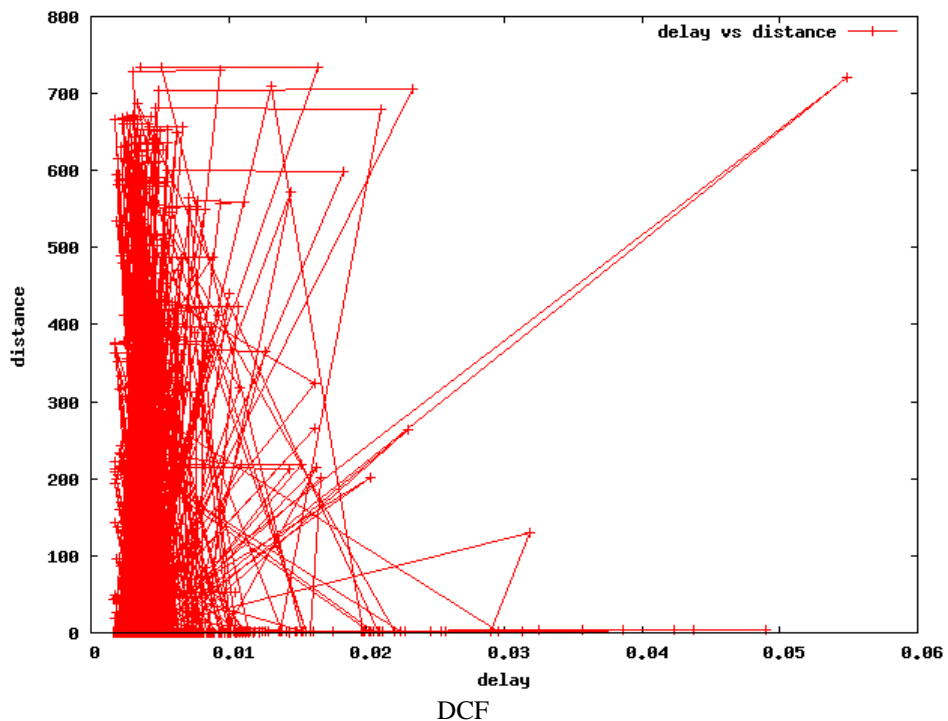


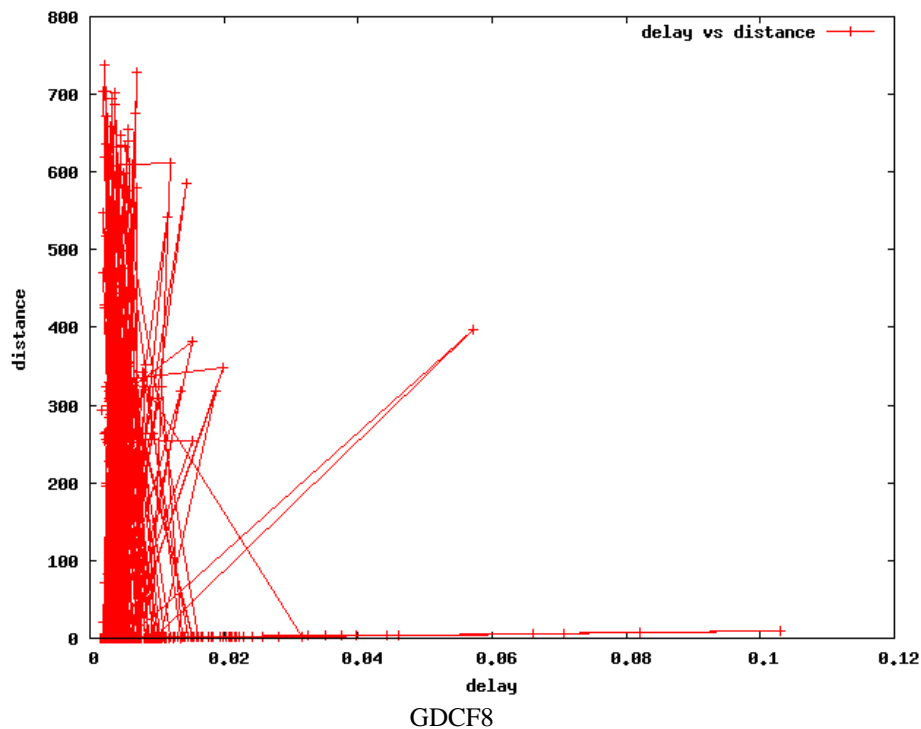
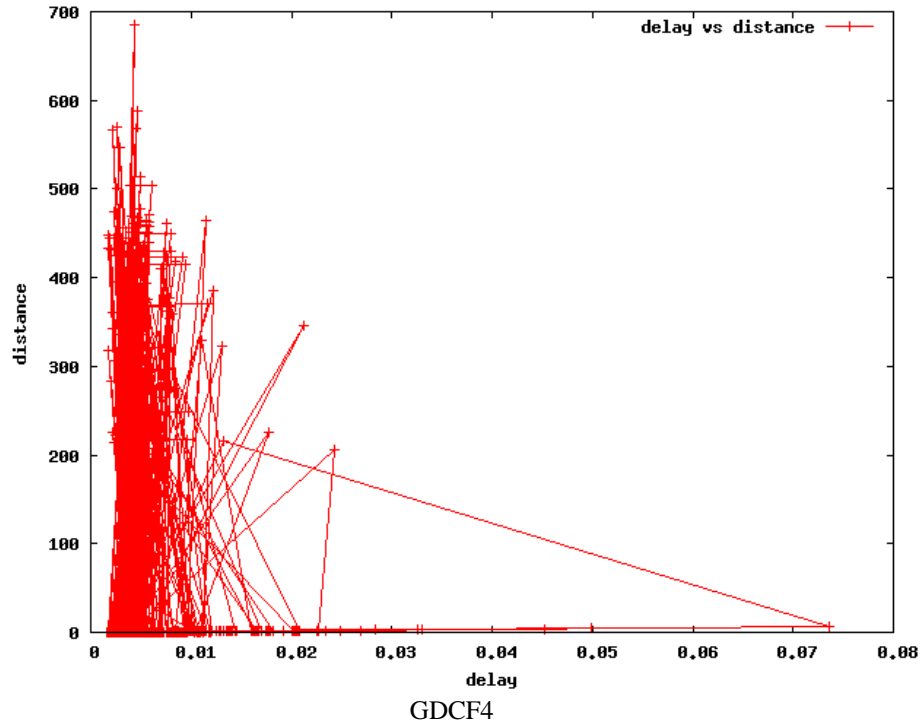
GDCF8

It can easily be seen from the graph that GDCF has significantly reduced the average End-to-End delay. As stated in GDCF, the optimization value  $c$  in GDCF will change according to different scenarios. In this experiment, I just use GDCF4 and GDCF8 as examples to show the performance of GDCF, even better results can be expected from GDCF.

Delay dependence on distance:

I get ns-scan file from Prof. Kim, and modify it to finish the simulation. I use Gnuplot to get those graphs. I set the transmission range of each node into 800m. Please see the performance results for DCF, GDCF4 and GDCF8.





Each point in the chart demonstrates an event. An event means a successful transmission of packet from one node to another, the X and Y value for that event is the delay of that transmission and the distance between those two nodes at that point. I just use this to demonstrate the stochastic relation between the delay of the transmission and the distance from the base station to the destination in that transmission.

It can be seen from the graph that the distance does not deteriorate the delay of the transmission. However, GDCF significantly lowers the delay rate of the whole simulation.

packet drop:

I then use ns-scan file to calculate the packet drop for each simulation, it can be seen from the table that the packet drop rate do not much difference between the DCF and GDCF.

Node number	Packet drop ratio		
	DCF	GDCF4	GDCF8
2	0.396072	0.396329	0.396586
3	0.446740	0.446484	0.450297
4	0.409250	0.466854	0.468531
5	0.392660	0.472623	0.472048
6	0.471436	0.473351	0.474979
7	0.476830	0.473810	0.476216
8	0.413700	0.472042	0.473751
9	0.447861	0.449548	0.414344
10	0.393244	0.333414	0.366650
11	0.445280	0.475337	0.444524
12	0.435751	0.438400	0.423711
13	0.403163	0.401864	0.415307

## Experiment 2: Network performance depending on the packet size in DCF and GDCF:

I change the packet size from 70 to 160 bytes, and tried to run the test script in DCF and GDCF (GDCF4 and GDCF8), in this experiment, I set the node number into 5, cbr rate 1mb.

### Mean throughput:

I use the same perl file to analysis the tracefile, and get this table. It appears that the packet size of the simulation is not appropriate. Using this kind of small packets will increase a lot of collisions.

Packet Size	DCF (bps)	GDCF4 (bps)	GDCF8 (bps)
70	2931254.9	2814208.69	2816934.91
80	2361173.94	2565450.66	2567504.99
90	2281669.34	2279757.68	2281339.26
100	2049680.75	2052234.07	1970881.74
110	1859440.45	1867786.77	1764257.17
120	1640173.22	1711350.62	1713269.55
130	1575969.92	1581563.35	1578677.1
140	1464930.7	1464847.31	1467396.32
150	1366888.09	1295603.63	1238023.66
160	1281038.21	1283960.06	1283917.83

### Mean Delay:

I use tracegraph to finish this part, please see the table below:

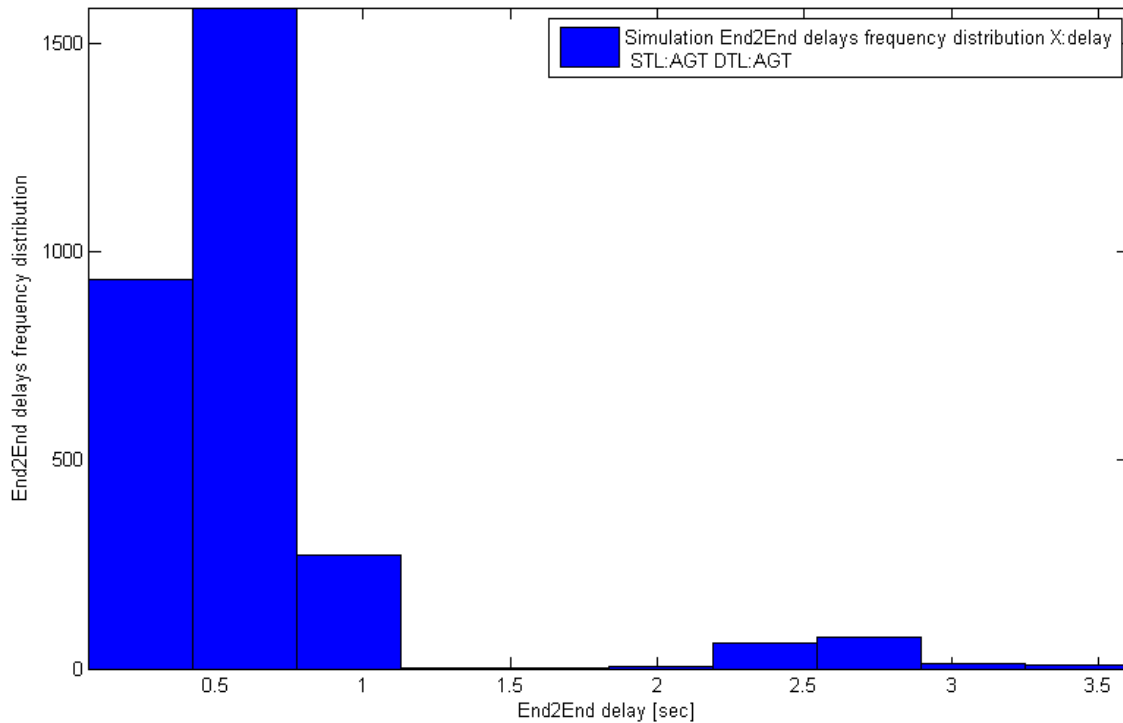
Packet Size	DCF	GDCF4	GDCF8
70	0.527944544	0.5000472014	0.468208095
80	0.5273692615	0.4560652708	0.4482777459
90	0.5612960854	0.4198352242	0.5127919252
100	0.5969917559	0.4138069198	0.4528887496
110	0.6104660394	0.4839410951	0.4724676396
120	0.6301883133	0.47691754	0.4472869939
130	0.6409891882	0.4137375443	0.4630992887
140	0.6558363337	0.5512090312	0.53408428
150	0.6706990961	0.4460792894	0.5296348376
160	0.6851888435	0.4437338215	0.4796519753

The data shows that the GDCF definitely has a lower mean delay than DCF. And as the packet size increase, the mean delay in DCF increase, however, in GDCF4 it actually decrease, and in GDCF8 it remains nearly the same. I

cannot know the optimization value of  $c$  in GDCF for each simulation, so this can only show a little of GDCF performance, but the superiority over DCF can be found definitely.

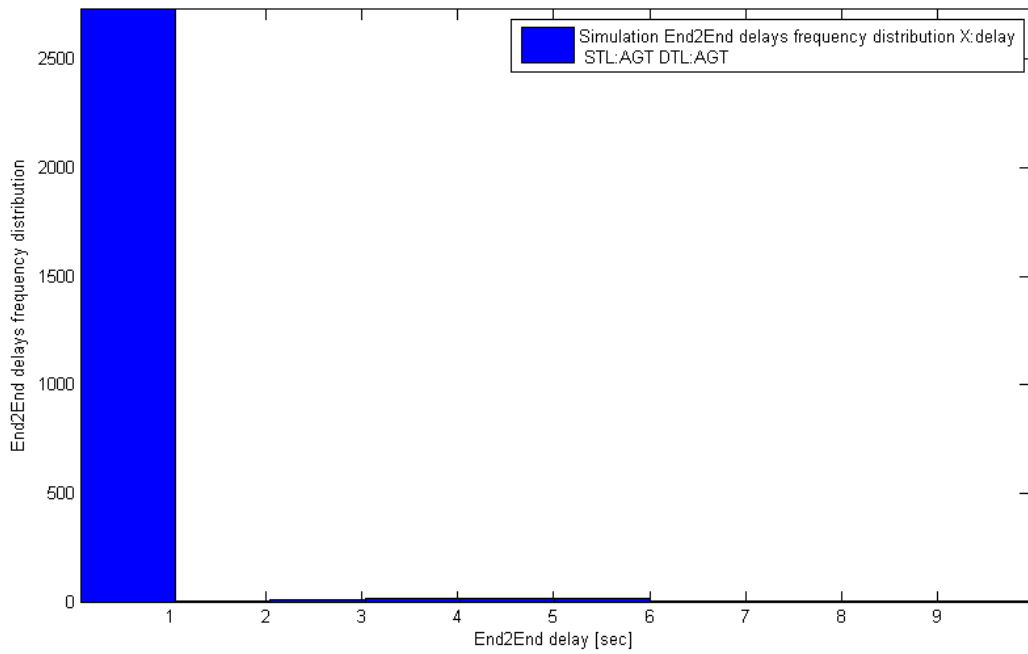
#### Delay statics:

I use tracegraph to analysis the tracefiles. All of the simulation chart show the same trend, so I choose packet 130 bytes as an example to show the difference of delay distribution in DCF, GDCF4 and GDCF8

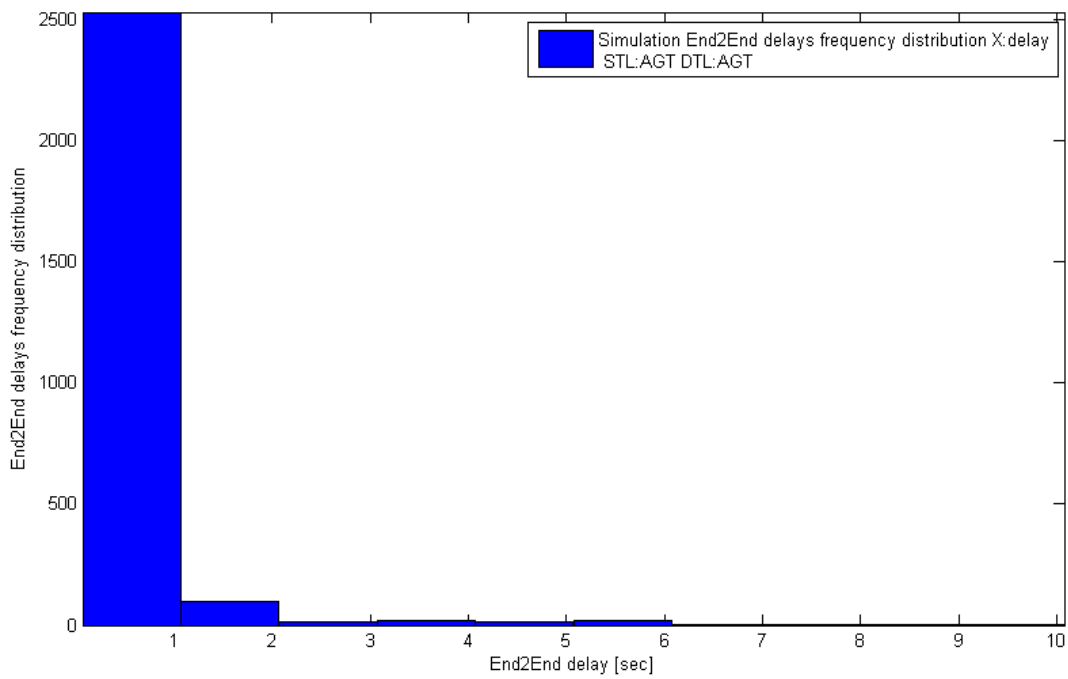


#### DCF

The DCF chart and the below GDCF chart show that the average delay definitely changed a lot. GDCF can help to make the majority of End-to-End delay come into small numbers. However, as the  $c$  is a optimal value, we cannot say anything about the performance of GDCF4 and GDCF8 in this simulation.



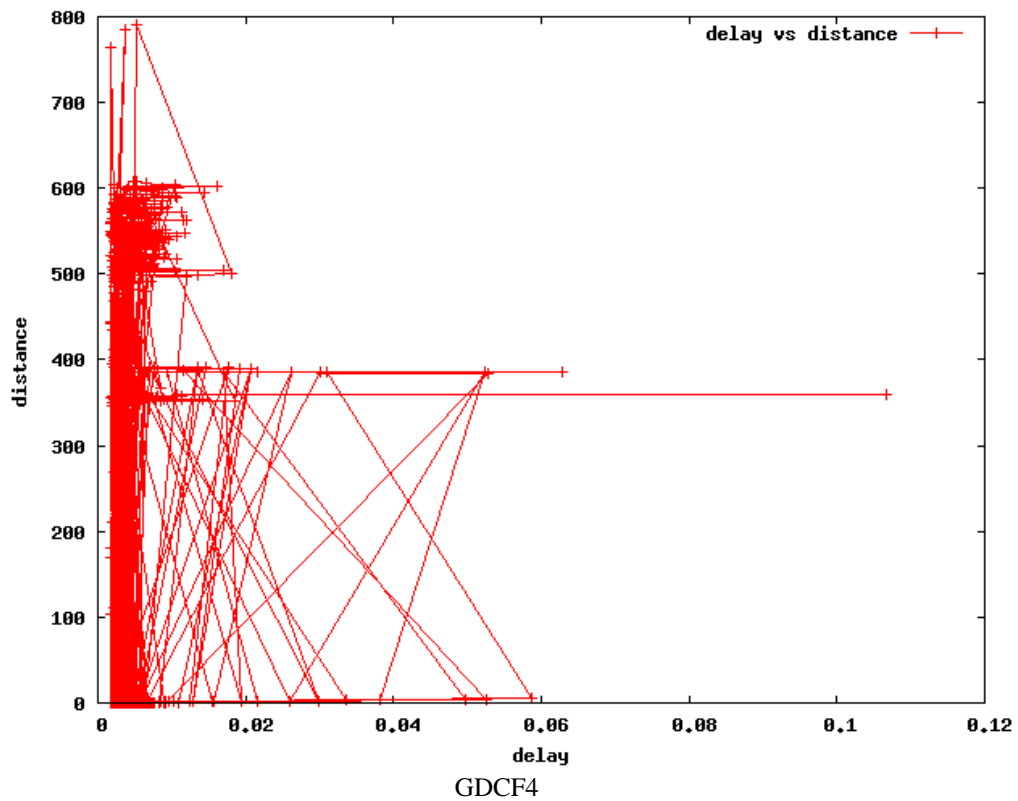
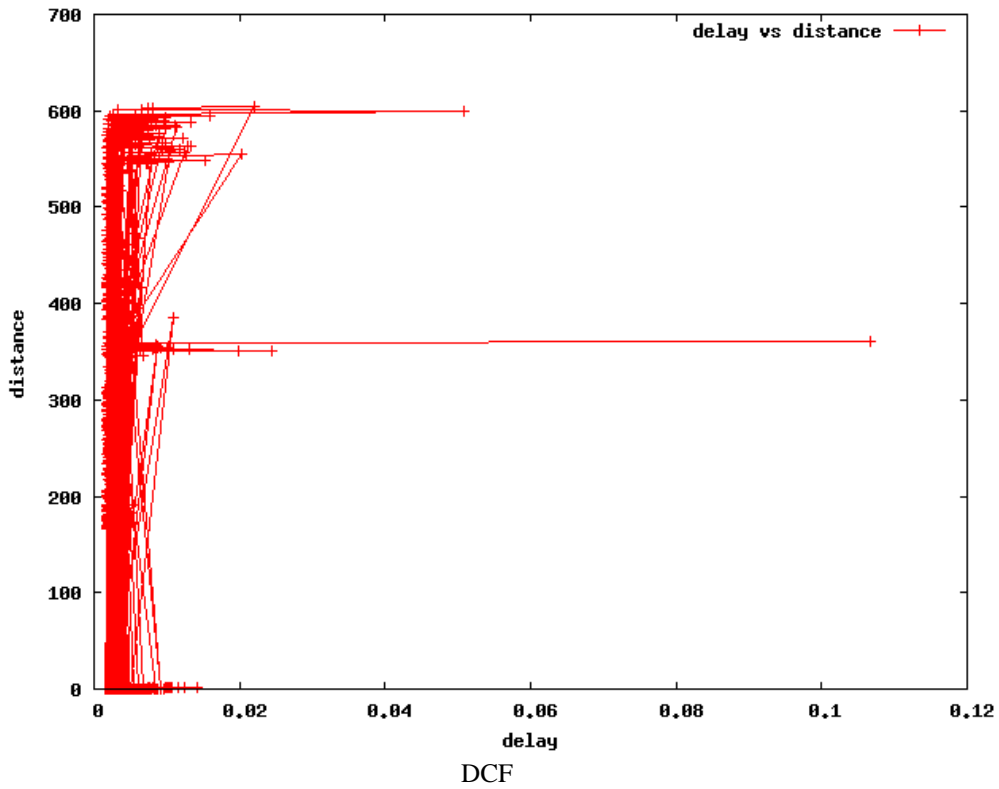
GDCF4

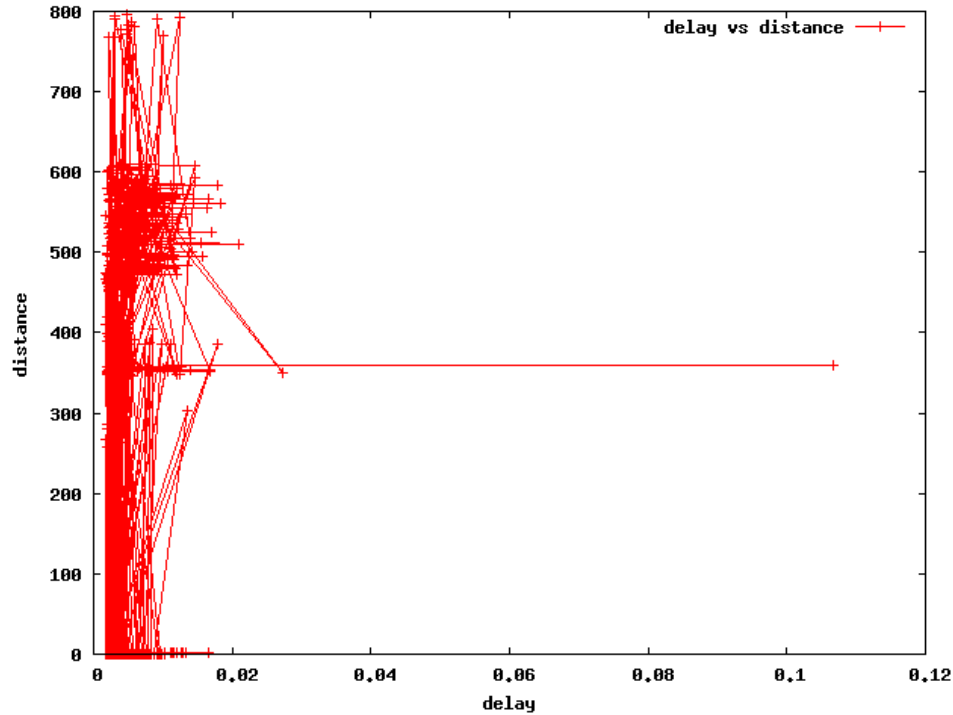


GDCF8

Delay dependence on distance:

I use ns-scan to finish obtaining all of the successful transmission data, I then use gnuplot to plot those data (for each data item, delay of the transmission and distance between the source node and destination node is what I want to plot out in this analysis).





GDCF8

From the graphs, we can see that different packet size does not help the GDCF to lower the delay. The majority of the delay remains the same stage as it is in DCF.

#### Packet drop:

I use ns-scan to finish calculating the packet drop ratio. It can be seen from here that the trend cannot be speculated from the data, may be the simulation still lacks enough number of different packet sizes to show the trend of GDCF and DCF, the obvious differences cannot be found.

Node number	Packet drop ratio		
	DCF	GDCF4	GDCF8
70	0.482621	0.436076	0.432988
80	0.373649	0.483918	0.484432
90	0.482581	0.482185	0.484281
100	0.482572	0.482386	0.433160
110	0.480819	0.480559	0.403736
120	0.431982	0.483131	0.482554
130	0.481549	0.481292	0.482860
140	0.482305	0.481891	0.479908
150	0.481481	0.402321	0.333766
160	0.481373	0.481145	0.481085

### III. CONCLUSION

In this paper, we have presented a performance analysis between the GDCF and DCF in order to prove the GDCF's superiority over DCF. Our model assumes a finite number of nodes in a small field and the ideal channel conditions. The test program is suited for any condition employed. Using the proposed test program, we have evaluated the 802.11 throughput performance and delay performance.

The problem lies in that the Random waypoint algorithm used in this simulation defines the nodes to move very fast, and the packet size of each transmission appears to be very small. Those shortcomings make more collisions to happen. And due to lack of hardware resources, I did not provide sufficient different scenarios to fully show the differences of DCF over GDCF.

## IV. REFERENCES

- [1] Hung. Fu-Yi, Marsic. Ivan, "Analysis of non-saturation and saturation performance of IEEE 802.11 DCF in the presence of hidden stations," IEEE Vehicular Technology Conference, 2007 IEEE 66<sup>th</sup> Vehicular Technology Conference, VTC 2007-Fall, 2007, pp. 230-234.
- [2] Guolin. Sun, Bo. Hu, "Delay analysis of IEEE 802.11 DCF with back-off suspension," Proceedings of the 15<sup>th</sup> International Conferences on Advanced Computing and Communications, ADCOM 2007, Proceedings of the 15<sup>th</sup> International Conference on Advanced Computing and Communications, ADCOM 2007, p 148-151, 2007.
- [3] Choi. Nakjunj, Seok. Yongho, Choi. Yanghee, Kim. Sungmann, Jung. Hanwook, "P-DCF: Enhanced backoff scheme for the IEEE 802.11 DCF," IEEE Vehicular Technology Conference, 2005 IEEE 61<sup>st</sup> Vehicular Technology Conference – VTC 2005 – Spring Stockholm: Paving the Path for a wireless Future, v 61, n 3, p 2067-2070, 2005.
- [4] Wang. Chonqqang, Li. Bo, Li. Lmin, "A new collision resolution mechanism to enhance the performance of IEEE 802.11 DCF," IEEE Transactions on Vehicular Technology, v 53, n 4, p 1235- 1246, July 2004.
- [5] Chung. Min Young, Kim. Min-Su, Lee. Tae-Jin, Lee. Yutae, "Performance evaluation of an enhanced GDCF for IEEE 802.11," IEEE Transactions on Communications, v E88-B, n 10, p 4125-4128, October 2005.
- [6] Giuseppe Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function, " IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 18, NO. 3, MARCH 2000

Feng An (06.01.1987- ) (fengan@iupui.edu) gets his Bachelor's Degree in Department of Microelectronics Science and Technology, School of Astronautics, Harbin Institute of Technology, Harbin, China, People's Republic of. in July 2008. He is currently with Department of Electrical and Computer Engineering, Purdue School of Engineering and Technology, Indiana University/Purdue University Indianapolis, pursuing a Master Degree in Electrical and Computer Engineering.

An has finished an internship program with Hongdu Aviation Industry Group, starting from July 15<sup>th</sup> to August 20<sup>th</sup> in 2007. An's research interests include VLSI, DSP, Mobile Wireless Network, Mobile Sensor Network, Microelectronics.

Prof. Dongsoo Kim (dskim@iupui.edu) is with Electrical and Computer Engineering Department, Purdue School of Engineering and Technology, Indiana University/Purdue University Indianapolis. He is a member of IEEE, a specialist in wireless networking and embedded systems.