ECE 478/578: Fundamentals of Computer Networks

Project # 1. The Distributed Coordination Function (DCF) of 802.11

1 Preliminaries

- Read the project description. When you are finished reading it, read it again.
- You can form a group of up to two people.
- You are free to use a programming language of your choice.

2 Project Description

You are to study the performance of multiple access protocols in a wireless setting. Consider the network shown in Figure 1. The circles denote the communication range R of each station. We are interested in the following two scenarios:

A. Concurrent Communications: Stations A, B, C, and D of Figure 1(a) are within the same collision domain (any transmission is received by all). Communication takes place between pairs $A \rightarrow B$ and $C \rightarrow D$. Traffic is generated at A and C according to a Poisson arrival process with parameters λ_A and λ_C, respectively.

B. Hidden Terminals: Stations A, B, and C of Figure 1(b), belong to two collision domains. Communication takes place between pairs $A \rightarrow B$ and $C \rightarrow B$. Traffic is generated at A and C according to a Poisson arrival process with parameters λ_A and λ_C, respectively.

For each scenario, compute relevant performance metrics for the following multiple access protocols. A time-slotted system is assumed.

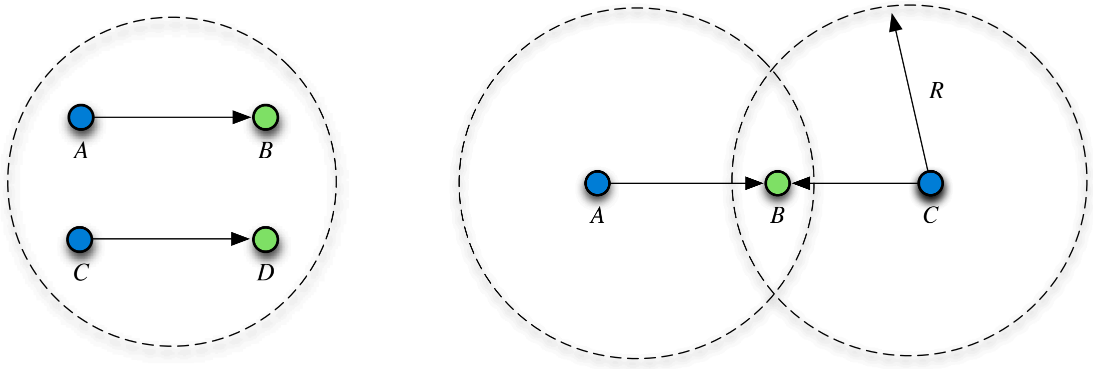
1. *CSMA with Collision Avoidance (CSMA/CA) according to the 802.11 DCF function.*

   (a) A station Tx ready to transmit (when a frame has arrived for transmission from the upper layers of the network stack) selects a random backoff value in $[0, CW_0 - 1]$. It first senses the channel for an initial period of DIFS time.

   (b) If the channel is busy, Tx (and every other station with a frame for transmission) monitors the channel until it becomes idle. When the channel becomes idle, Tx decrements his counter by one with every idle slot. If the channel becomes busy, Tx freezes its backoff counter. When the counter reaches zero, Tx transmits its frame.

   (c) If the frame is successfully received (no collision) by Rx, the station Rx replies with an ACK frame after SIFS time. This completes the transmission round and the protocol repeats for the next transmission. For successive transmissions, the station has to sense for DIFS time before starting the countdown.

   (d) If a collision occurs, the stations that collided double their contention window CW and repeat the backoff process. After k collisions, the backoff value is selected from $[0, 2^k CW_0 - 1]$. The CW value cannot exceed threshold $CW_{\max}$.



(a) (b)

Figure 1: (a) Topology for parallel transmissions within the same collision domain, (b) topology for parallel transmissions when A and C are hidden terminals.

2. *CSMA/CA with virtual carrier sensing enabled:* RTS and CTS frames are exchanged before the transmission of a frame. If RTS transmissions collide, stations invoke the exponential backoff mechanism outlined in 1(d). Otherwise, stations that overhear an RTS/CTS message defer from transmission for the time indicated in the NAV vector.

3 Simulation parameters

Parameter	Value	Parameter	Value
Data frame size	1,500 bytes	ACK, RTS, CTS size	30 bytes
Slot duration	10 μs	DIFS duration	40 μs
SIFS duration	10 μs	Transmission rate	12 Mbps
CW_0	4 slots	$CW_{\max}$	1024 slots
λ_A, λ_C	$\{50, 100, 200, 300, 500\}$ frames/sec	Simulation time	10 sec

4 Performance Metrics

Evaluate the protocol performance with respect to the following metrics:

Throughput T: The individual station's throughput as a function of λ.

Collisions N: The number of collisions (data and RTS/CTS) as a function of λ.

Fairness Index FI: The fraction of time that the channel is occupied by pair $A \rightarrow B$ over the fraction of time that the channel is occupied by pair $C \rightarrow D$ as a function of λ.

Your experiments must be repeated for two different scenarios: (a) $\lambda_A = \lambda_C = \lambda$ and (b) $\lambda_A = 2\lambda, \lambda_C = \lambda$. Assume no losses due to the imperfections of the wireless medium.

5 Project Report

Include with your report:

- A brief introduction describing the project. In this section, include the responsibilities of each team member with references to which parts/steps he/she completed.

- A description on how you developed your simulations.
- Graphs for each of the simulated scenarios.
  1. Throughput T
     (a) **Node A:** Throughput T (Kbps) vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2 (four lines in total).
     (b) **Node C:** Throughput T (Kbps) vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2 (four lines in total).
     (c) **Node A:** Throughput T (Kbps) vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2, when $\lambda_A = 2\lambda_C$ (four lines in total).
     (d) **Node C:** Throughput T (Kbps) vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2, when $\lambda_A = 2\lambda_C$ (four lines in total).
  2. Collisions N
     (a) **Node A:** Number of collisions N vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2 (four lines in total).
     (b) **Node C:** Number of collisions N vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2 (four lines in total).
     (c) **Node A:** Number of collisions N vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2, when $\lambda_A = 2\lambda_C$ (four lines in total).
     (d) **Node C:** Number of collisions N vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2, when $\lambda_A = 2\lambda_C$ (four lines in total).
  3. Fairness Index FI
     (a) Fairness Index FI vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2 (four lines in total).
     (b) Fairness Index FI vs. rate λ (frames/sec) for scenarios A and B, and CSMA implementations 1 and 2, when $\lambda_A = 2\lambda_C$ (four lines in total).
- **Justification for the results shown in your graphs. The majority of your grade will be based on your interpretation of the results, not the results themselves, so spend adequate time explaining them.**

Appendix

Generating Poisson-distributed traffic: To generate Poisson-distributed traffic, it is sufficient to generate a series of exponentially-distributed inter-arrival times. Such times can be generated using the inverse CDF transformation method.
Step 1: Generate a series of uniformly distributed numbers $U = \{u_1, u_2, \ldots, u_n\}$ with $u_i \in (0,1), \forall i$.
Step 2: Compute series $X = \{x_1, x_2, \ldots, x_n\}$ of exponentially distributed numbers with λ, as

$$X = -\frac{1}{\lambda}\ln(1-U) \tag{1}$$

Using X, you can determine the time of each frame arrival at each station. For instance, frame 1 arrives at time x_1, frame 2 arrives at time $x_1 + x_2$, etc. The inter-arrival time generation process has to be repeated for each transmitting station.

Plotting tips:

1. Label your axes and use appropriate units
2. Make sure the scales on both axes are appropriate. If you are to use the same variable on multiple plots (e.g. throughput) use the same scale on all plots so they can be compared
3. Do not superimpose more than 4-5 plot lines on the same plot.
4. If more than one plot lines are present in the same plot make sure to individually label each one
5. For individual plot lines use different marker shapes so they can be distinguishable.
6. Keep in mind that colors do not show on a black and white printout. So if you color code your lines, use some other discernable labeling such as dashed lines to differentiate between plot lines.

MATLAB code for generating good figures. You can port your data to Matlab and use this code to generate figures.

```
close all; % closes all open figure windows

set(0,’defaulttextinterpreter’,’latex’); % allows you to use latex math
set(0,’defaultlinelinewidth’,2); % line width is set to 2
set(0,’DefaultLineMarkerSize’,10); % marker size is set to 10
set(0,’DefaultTextFontSize’, 16); % Font size is set to 16
set(0,’DefaultAxesFontSize’,16); % font size for the axes is set to 16

figure(1)
plot(X, Y1, ’-bo’, X, Y2, ’–rs’, X); % plotting three curves Y1, Y2 for the same X
grid on; % grid lines on the plot
legend(’CSMA’, ’CSMA w. virtual Sensing’);
ylabel(’$T$ (Kbps)’);
xlabel(’$λ$’ (frames/sec));
```