Expt II: IEEE 802.11 WLAN: Distributed Coordination Function (DCF)

I. Goals

1) Understand basics of DCF protocol (CSMA/CA);
2) Use ns-3 to simulate IEEE 802.11 DCF WLAN network performance

Your repository should be at changeset 570961abd or later (Friday, February 8).

II. IEEE 802.11 Wireless Local Area Network

IEEE 802.11 WLANs (commonly known as WiFi) have gained widespread success and popularity over the past decade.

Fig. 1 illustrates the basic architecture of IEEE 802.11 WLANs. The fundamental component of an IEEE 802.11 WLAN is the Basic Service Set (BSS) - the equivalent of a ‘cell’ in cellular networks. A BSS contains one or more wireless stations, such as computers or mobile phones, and an access point (AP) cabled to the wired backbone network to allow the wireless stations access to the Internet. The BSSs represent an ‘infrastructure mode’; the wireless stations can also group themselves together to form an ad-hoc network without any infrastructure (AP). In the ad-hoc mode, each node can communicate with other nodes without any backhaul to the Internet.

A. IEEE 802.11 Standard Family

Since the U.S. FCC released the ISM band for unlicensed use in 1985, a number of IEEE 802.11 standards have been proposed by the IEEE LAN/MAN Standards Committee (IEEE 802.11) to implement WLAN communication, which are summarized in Table I.

The original IEEE 802.11 standard, released in June 1997 is obsolete today. This standard operated at 2.4 GHz and specified two data rates of 1 Mbps and 2 Mbps. Three alternative physical layer technologies were implemented, including diffuse infrared, Frequency Hopping Spread Spectrum (FHSS) and Direct Sequence Spread Spectrum (DSSS). The 802.11b standard was later released in Sept. 1999 as a direct

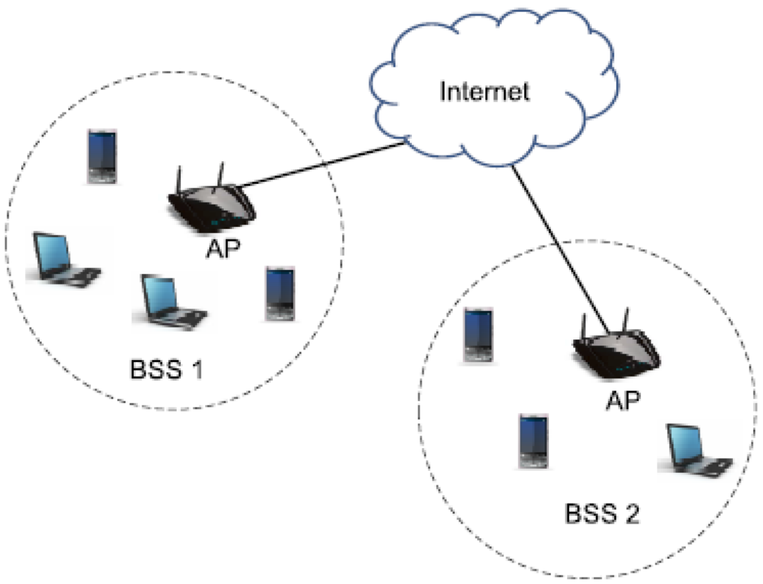

Fig. 1. IEEE 802.11 WLAN architecture

TABLE I
IEEE 802.11 STANDARDS

Protocol	Release	Freq.(GHz)	Data Rate(Mbps)	Physical Layer	Bandwidth(MHz)
802.11-1997	June 1997	2.4	1, 2	DSSS, FHSS	20
a	Sept. 1999	5	up to 54	OFDM	20
b	Sept. 1999	2.4	up to 11	DSSS	20
g	June 2003	2.4	up to 54	DSSS, OFDM	20
n	Oct. 2009	2.4, 5	up to 600	MIMO, OFDM	20, 40
ac	Jan. 2014	5	up to 3400	MIMO, OFDM	up to 160

extension of the legacy 802.11 but with a higher maximum data rate of 11Mbps. At the same time, the 802.11a standard was proposed that operated at the less-crowded 5 GHz band with a maximum data rate of 54 Mbps. It also used a more advanced air interface technology or PHY layer, based on Orthogonal Frequency Division Multiplexing (OFDM). 802.11g, ratified in June 2003 uses the same OFDM-based PHY layer as 802.11a (maximum data rate of 54 Mbps) but for 2.4 GHz.

The peak physical layer data rate was further boosted in Oct. 2009, with 802.11n or Multiple-Input Multiple-Output (MIMO). It supported both 2.4 GHz and 5 GHz bands with data rates from 54 Mbps to 600 Mbps with channel bandwidths of 20 & 40 MHz. In the 802.11ac standard released in Jan. 2014, the peak rate grew upto 1.3 Gbps by using even wider channel bandwidth (including 80 & 160 MHz), more streams (up to eight), and higher-density modulation (up to 256 QAM).

B. Distributed Coordination Function

The IEEE 802.11 MAC layer defines two types of mechanisms: a mandatory Distributed Coordination Function (DCF) and an optional Point Coordination Function (PCF). PCF is a centralized scheme that provides contention-free channel access but is rarely implemented in current products due to its complexity and will be not be considered further.

DCF is the fundamental MAC protocol of the IEEE 802.11 based WLAN standard. It adopts the CSMA/CA with Binary Exponential Backoff (BEB) algorithm, along with two access mechanisms, a two-way handshaking scheme called basic access and a four-way handshaking scheme called request-to-send/clear-to-send (RTS/CTS) access. In the following, the core operational steps in DCF will be summarized.

Suppose that a node (a mobile station or an AP) has a packet to transmit. With the basic access mechanism:

1) If the node senses the channel idle, it transmits the packet after determining the channel is continuously idle for a duration of time known as Distributed Interframe Space (DIFS).
2) If the channel is sensed busy, the node waits until the channel is sensed idle for a duration of DIFS. The node then chooses a random backoff value from a backoff window $\{0,...,CW_{\min}\text{-}1\}$, where $CW_{\min}$ **is referred to as the initial backoff window size**. It counts down this value at each subsequent idle time slot [1], freezes the count-down process anytime the channel is sensed busy, and reactivates it when the channel is sensed idle for a duration of DIFS.
3) When the counter reaches zero, the node transmits its packet and then waits for an acknowledgement (ACK). If the ACK is received within a duration of Short Interframe Space (SIFS), the transmitting nodes knows that its packet has been successfully received by the destination. If the node has another packet to transmit, it re-starts the DCF operation at Step 2.
4) If the ACK is not received within a specified time interval called ACK_Timeout or it senses the transmission of another packet on the channel, it restarts the backoff process at Step 2 with the random backoff value chosen from an increased backoff window, where the backoff window size is doubled (hence, called the Binary Exponential Backoff algorithm).

Fig. 2 illustrates an example of the operation of DCF with the basic access mechanism. Two nodes A and B share the wireless channel. As can be seen from Fig. 2, during the transmission of node A,

[1]A time slot is the basic unit of timing in 802.11 WLAN, equal to 9 μs. in our case

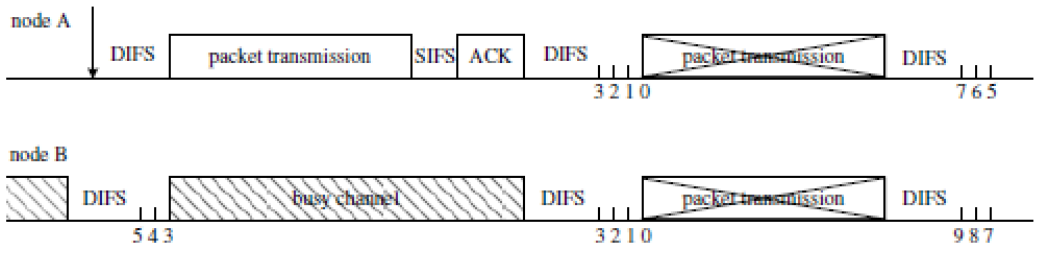

Fig. 2. Graphic illustration of successful transmission and collision in IEEE 802.11 DCF networks with basic access mechanism

node B senses the channel busy and suspends the counting-down process of its backoff counter. When the channel is sensed idle for a duration of DIFS, node B resumes to decrease its backoff counter. If node A and node B attempt to access the channel at the same time, a collision occurs, whereby both the participating frames are assumed to be un-decodable at the intended receiver. After such a collision, both nodes double their current backoff window value and reenter the backoff process. A packet may experience a maximum number of consecutive collisions; K **denotes the maximum # of retransmission stage**. Thereafter for subsequent collisions, the backoff window size no longer doubles and stays at its maximum value $CW_{\max} = CW_{\min} \cdot 2^K$ upto a *retry limit*; if the packet is still not successful, it is dropped and the window size reverts to $CW_{\min}$.

The RTS/CTS access mechanism is an optional feature within DCF, introduced to deal with the so-called ‘hidden-terminal’ problem in IEEE 802.11 WLANs. RTS/CTS access mechanism allows a node to use Request-to-Send (RTS) and Clear-to-Send (CTS) control frames to reserve access to the channel as follows.

i. When a node has a packet to transmit, it first sends an RTS frame to the destination, which contains the total time interval required to transmit the entire remaining packet exchange (through the reception of the ACK).

ii. When the destination receives the RTS frame, it responds by broadcasting a CTS frame after a duration of SIFS, which gives the requesting node the permission to send and also notifies the other nodes in the vicinity not to transmit during the reserved time interval (which again spans through the reception of the ACK).

iii. Any listening node in the network that hears the information is able to update the Network Allocation Vector (NAV) with the reservation time duration, and will not attempt to access the channel during the time period that the NAV indicates to be occupied. The packet transmission starts after the successful exchange of the RTS and CTS frames, and is also confirmed to be successful by the ACK frame. An illustration of the RTS/CTS access mechanism is shown in Fig. 3.

Note: The experiments will ONLY explore the Basic Access DCF performance and *NOT* the use of RTS/CTS.

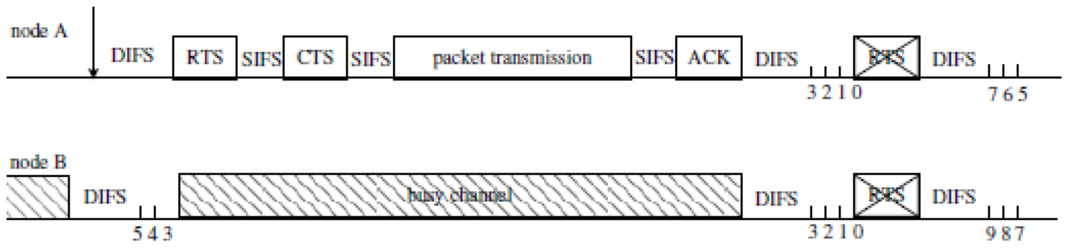

Fig. 3. Graphic illustration of successful transmission and collision in IEEE 802.11 DCF networks with RTS/CTS access mechanism

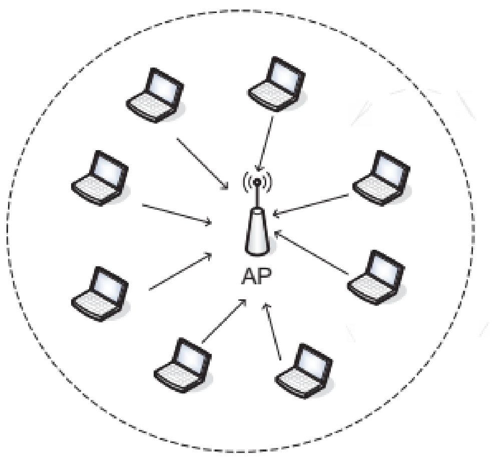

Fig. 4. Illustration of network topology.

III. NS-3 Experiment: Performance Evaluation of DCF Basic Access

We will explore performance evaluation of the DCF Basic access protocol for various scenarios by adjusting network parameters: number of network nodes n, the initial backoff window size $CW_{\min}$, the max. retransmission stage K and observing their impact on the network (aggregate) and per-user throughput or delays, to deduce trends and relationship between input network parameters and the observations.

A. Simulation Setup

Consider a single IEEE 802.11 DCF network where n nodes transmit to an access point (AP); the network topology is shown in Fig. 4. The nodes are placed randomly within the transmission range of the AP, for a *fixed* data rate DR. The carrier sensing range is set larger than the transmission range, and hence all transmissions can be detected by any node (implying no 'hidden' terminals).

A packet transmission is successful if and only if there are no concurrent transmissions; otherwise, a collision occurs and none of the packets can be successfully decoded. In the script, we use SpectrumWifi helpers for setting wifi channel configuration. The propagation delay is set according to $distance/c$ where c is the speed of light) and a Friis path loss model is configured (both these have little effect on the results). We set the duration of simulation to be a configurable duration, and the latter $duration - 1$ seconds are used for the throughput calculation.

```
ns3.29-wifi-dcf-debug [Program Options] [General Arguments]
Program Options:
    --ascii:         Print ascii trace information if true [false]
    --pcap:          Print pcap trace information if true [false]
    --animate:       Print animation trace if true [false]
    --packetSize:    Set packet size (bytes) [1900]
    --packetArrivalRate:  Packet arrival rate per second [1]
    --numStas:       Number of STA devices [1]
    --cwMin:         CwMin parameter of DCF [15]
    --cwMax:         CwMax parameter of DCF [1023]
    --duration:      Duration of data logging phase (s) [10]
    --radius:        Radius for node dropping around AP (m) [25]
    --rtsThreshold:  Packet size threshold for RTS (bytes) [2200]
```

Please note the `--rtsThreshold` option, which is available for people to experiment with enabling the RTS/CTS exchange. The questions below do not require use of this parameter. The default setting of 2200 bytes is sufficient to disable it because all packets we will use will be smaller than 2000 bytes.

B. An Example: Network Throughput versus User Traffic Arrival Rate

To adjust the traffic arrival rate of each node, the relevant snippet in the source code is given below:

```
$ ./waf --run 'wifi-dcf --packetArrivalRate=100'
```

When specifying a packet arrival rate, this rate will be used for each STA in the network (i.e. not split among the total population of STAs).

Run the simulation for distinct packet arrival rates, from 10 to $10,000$), obtain the corresponding network throughput and plot throughput vs. packet arrival rate, as illustrated in Fig. 5. You can obtain the total throughput by counting the successful arrivals in the file `wifi-dcf-ap-rx-trace.dat`, converting the packet size to bits, and dividing the total bits by the duration of the simulation (defaulting to 10 seconds in this case). The program actually does this for you (check the output printed upon completion). We can see that the network throughput grows linearly with the traffic arrival rate at first, and subsequently becomes insensitive to the arrival rate when it is higher than 3000 packets per second. Note also that this saturates well below the maximum rate of 54 Mb/s (due to the overhead involved with the sequence of Data + Ack + DIFS + backoff for each frame– this is the problem that Wi-Fi aggregation addresses).

When the user traffic arrival rate is small (arrival interval is large), the network is unsaturated (queues may be empty at times) and the throughput increases in proportion to the traffic arrival rate. When the user traffic arrival rate grows to a critical value, each node always has a packet to transmit (network is "saturated"), the throughput reaches a steady-state value, which will no longer change with the increase of the traffic arrival rate.

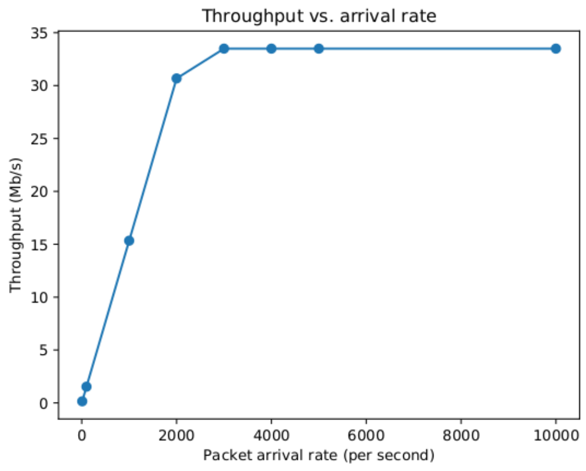

Fig. 5. Simulated network throughput versus the traffic arrival rate of a single node

The above provides a generic example for your actual ns-3 assignments. The basic steps are to

1) Run simulations for varying values of a system parameter (packet arrival rate), and obtain the corr. network throughput for each value (as found in the AP received packet trace or printed out by the program);

2) Plot a figure to illustrate the relationship between network throughput and the system parameter;

3) Explain reasons for your observations.

IV. Experiment 1: Network saturation throughput versus Number of Nodes

To explore the above, simulate a saturated network with the packet length of 1900 bytes (the default), the initial backoff window size of $CW_{\min} = 15$ (the default), the max. retransmission stage $K = \log(CW_{\max}/CW_{\min}) = 6$ (the default), the data frame transmission rate of $DR = 54$ Mbps (the default), and the control frame transmission rate of $BR = 24$ Mbps (the default).

Let us clarify the notation K. The minimum backoff window size is $CW_{\min}$, which will be a power of 2, minus 1 (e.g. 15). The maximum is $CW_{\max}$ which is also a power of 2, minus 1. If we add 1 to both quantities, and divide $CW_{\max}/CW_{\min}$ we get a value (64) and we take the base 2 logarithm of that value (i.e., 6). K is therefore the number of times we will double CW until we reach the maximum.

The network can't possibly handle more than 4000 packets per second, no matter the number of nodes. Set the packet arrival rate to be $4000/n$ for number of nodes n. Ten seconds (default) is enough simulation time to observe the effect (although you can confirm this by trying longer simulation times and seeing if your results change).

Vary the number of network users (5, 10, 15, 20, 25, 30, 35, 40) and obtain the corresponding network saturation throughput via simulation as follows.

```
./waf --run 'wifi-dcf --numStas=5 --packetArrivalRate=800'
./waf --run 'wifi-dcf --numStas=10 --packetArrivalRate=400'
... (etc.)
```

Answer the following questions:

1) Plot a figure of throughput vs. number of nodes n.
2) According to the figure in step 1), suggest a reason why the throughput decreases with the number of nodes in the network.
3) For IEEE 802.11 DCF networks, **collisions** occur when two or more nodes transmit their packets simultaneously. Identify any collisions in your trace file `wifi-dcf-rx-error-trace.dat`. How do the number of collisions increase with network size?

V. Experiment 2: Network Throughput versus Initial Backoff Window Size

To explore the above, simulate a saturated network with the packet length of 1900 bytes, number of nodes $n = 20$, the max. retransmission stage $K = \log(CW_{\max}/CW_{\min}) = 6$, the packet transmission rate of $DR = 54Mbps$, and the control frame transmission rate of $BR = 24Mbps$.

The initial backoff window size $CW_{\min}$ is sequentially set to 3, 7, 15, 31, 63, 127, 255, 511, 1023; the corresponding $CW_{\max}$ is set to 255, 511, 1023, 2047, 4095, 8191, 16383, 32767, 65535.

Use the `--cwMin` and `--cwMax` command line arguments to vary $CW_{\min}$ and $CW_{\max}$, as follows:

```
./waf --run 'wifi-dcf --numStas=20 --packetArrivalRate=200 //
    --cwMin=15 --cwMax=1023'
./waf --run 'wifi-dcf --numStas=20 --packetArrivalRate=200 //
    --cwMin=31 --cwMax=2047'
... (etc.)
```

For each $CW_{\min}$, run the simulations and obtain the corresponding simulated throughput.

Answer the following questions:

1) Plot throughput vs. initial backoff window size $CW_{\min}$.
2) Explain reasons for the figure in step 1), i.e. how the network saturated throughput varies with the initial backoff window size $CW_{\min}$.
3) What is the simulated value of $CW_{\min}$ that maximizes the network throughput?
4) Now increase the number of nodes n to 30 and 40. Rerun the simulations and obtain the corresponding best values of $CW_{\min}$. How does the best value change with the network size (if at all)?

VI. Experiment 3: Network Throughput versus Max. Retransmission Stage K

Simulate a saturated network with the packet length of 1900 bytes, number of nodes $n = 20$, the initial backoff window size $CW_{\min} = 15$, and again, the packet transmission rate of $DR = 54Mbps$, and the control frame transmission rate of $BR = 24Mbps$.

Set the max. retransmission stage K is set to $2, 4, 6, 8, 10, 12, 14, 16, 18, 20$, i.e., the attribute of $CW_{\max}$ is sequentially set to $CW_{\min} \cdot 2^K$.

For each max. retransmission stage K, run the simulations and obtain the corresponding simulated throughput.

Answer the following questions:

1) Plot throughput vs. the max. retransmission stage K.
2) According to your results in step 1), describe how the network saturated throughput varies with the max. retransmission stage K and suggest reasons (e.g. why the network throughput will eventually become insensitive to K?)

3) Compare with results in Section V and explain the difference between effects of $CW_{\min}$ and K on throughput.

VII. Experiment 4: Fairness in IEEE 802.11 DCF

Simulate a saturated network with the packet length of 1900 bytes, the initial backoff window size of $CW_{\min} = 15$, the max. retransmission stage $K = \log(CW_{\max}/CW_{\min}) = 6$, the packet transmission rate of $DR = 54$ Mbps, and the control frame transmission rate of $BR = 24$ Mbps. Run the simulation for number of nodes $n = 10$, and obtain your trace file.

Note: At each time slot, the channel can be in two states, *Idle* (no transmission) and *Busy* (channel is occupied by a transmission). A *Busy* state may result in either a successful transmission or a collision.

The `wifi-dcf-state-trace.dat` trace file shows, for each node, the time that it initiates a transmission, and its duration; e.g.

```
8.196788 0 state: TX start: 8.196788 duration 0.000028
```

These times (start and duration, units of seconds) can be used to determine the idle and busy times of the channel.

Answer the following questions:

1) During the entire simulation time, what fraction of time is the channel idle (busy)? From the perspective of a single node (e.g., node 0), sum the total idle times observed to determine the channel idle time.
2) Take a closer look at the busy period of the channel, which consists of transmissions from all the nodes. For Node $\#1$ to $\#10$, obtain the sum of transmit times of each node and list them in a table. From the table, comment on the access fairness of the DCF protocol.
3) Busy period leads to successful transmissions and collisions. For each Node $\#1$ to $\#10$, count the number of successful transmissions and collisions of each node and list them in a table. What is the probability of success for each node?
4) Increase the number of nodes from 10 to 20, 30, and 40, and re-run the simulations. Plot a figure to see how the idle and busy time fraction vary with the number of nodes n.
5) Similar to step 4), plot a figure to see how the average number of successful transmissions and collisions per node vary with the number of nodes n.