Question 1)

The expected number of slots that my node will wait before sensing the medium will be 4.

Question 2)

If the backoff exponent is 2, then number of states that the node will have is 4.

At the end of time slot 1, -70 dBm is detected. Since the CCA threshold is equal to -75 dBm, the channel is busy and the first number, 1, from the RNG sequence is generated. The packet is not sent because the power level that the node sensed is below the threshold.

At the end of time slot 2, the first retrial opportunity is used. -65 dBm is sensed and, as a result, the channel is busy. Therefore, the packet is not sent. The second number, 2, from the RNG sequence is generated.

At the end of time slot 3, the second retrial opportunity is used. -74 dBm is sensed and, since it is below the threshold, the channel is busy. Resulting in the packet not being sent. The third number, 2, from the RNG sequence is generated.

At the end of time slot 4, the third retrial opportunity is used. -90 dBm is sensed and, finally, the channel is free. Therefore, the packet is sent. The backoff counter is decremented. Therefore, the slot is moved from number two to number one. It starts at number two due to the RNG sequence number, 2. Then, from number one to zero. After backoff counter is set to zero, the node will sense again.

At the end of time slot 5, a second packet will be sent. -95 dBm is sensed and the channel is free because the value is above the threshold. Since the channel is free, the RNG does not generate a number and the packet is sent.

Question 3)

Percentage of packets received at the receiver: 18%

Average backoff length at the sender: 416.9

Question 4)

Percentage of packets received at the receiver: 47%

Average backoff length at the sender: 621.2

Question 5)

When the backoff length is larger, we received a larger amount of packets. Therefore, the average backoff length at the sender is directly proportional to the percentage of packets received at the receiver.

Question 6)

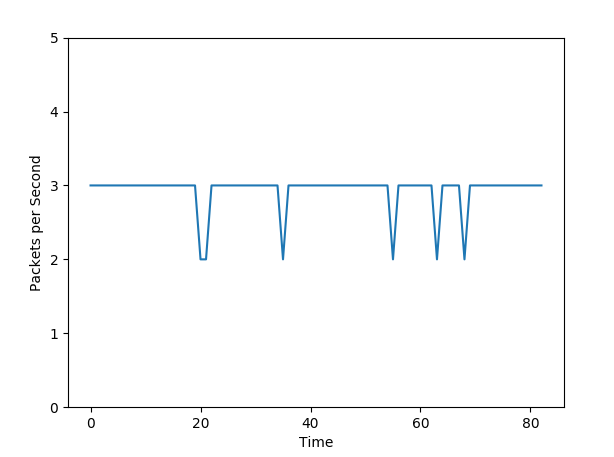
Percentage of packets received at the receiver: 37%

Average backoff length at the sender: 6630

Question 7)

According the results I got from the laboratory experiments, the best set of parameters that had the best results in networks with external interference are small backoff exponent, equal to 2, and low CCA offset, -95 dBm. I obtained such results because the throughput reduces if the backoff exponent increases. This is due to the fact that it will take a longer time to get the counter back to state zero, since it is backing up more. Furthermore, if the CCA offset is reduced then it will be able to detect lower quantities of power signals even if the distance is increased between the sender and the receiver. Thus, the optimal point is a low CCA offset and small backoff exponent.

Question 8)



Question 9)

import pyshark

import matplotlib.pyplot as plt

directory = "./Lab4Data/rasp11/"

filename\_txtlist = ["exp5.1.txt", "exp5.2.txt", "exp5.3.txt", "exp5.4.txt","exp5.5.txt", "exp5.6.txt"]

filename\_pcaplist=["exp5.1.pcap", "exp5.2.pcap","exp5.3.pcap", "exp5.4.pcap", "exp5.6.pcap"]

txtfilename = directory + filename\_txtlist[3]

file= open(txtfilename, 'r')

accum=0

#read first line

line= file.readline()

while line:

   if "csma: backoff" in line:

           # Split my string using space separator

           substrings = line.split(" ")

           # Get the value

           value = int(substrings[2])

           accum += value

#print 'accum=', accum

   # Read the next line

   line = file.readline()

#pcap data processing

pcpfilename = directory + filename\_pcaplist[3]

# Open the file

capfile = pyshark.FileCapture(pcpfilename)

init\_time=0

counter=0

cnt=0

packets=[]

for pkt in capfile:

   try:

       if pkt.wpan.fcs\_ok == '1':

           src\_addr = int(str(pkt.wpan.src16), 16)

           #print "Packet received from " + str(pkt.wpan.src16)

           if src\_addr == 0xed2e:

               counter+=1

               cnt+=1

               curr\_time= float(pkt.sniff\_timestamp)

               if cnt==1:

                       init\_time=curr\_time

               elif curr\_time>=(init\_time+1):

                       packets.append(cnt)

                       cnt=0

               #print 'counter:', counter

   except:

       pass

#print 'final accum=', accum

#print 'final counter=', counter

time\_list=range(0,len(packets))

plt.ylabel('Packets per Second')

plt.xlabel('Time')

plt.ylim(0,5)

plt.plot(time\_list, packets)

plt.show()

plt.savefig('ALplot8.png')

percent=counter

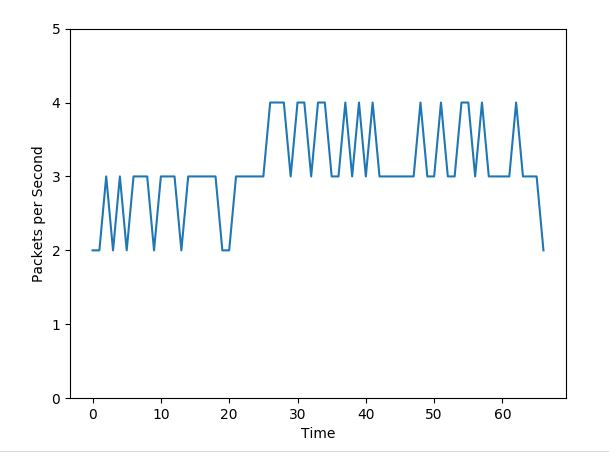
average = float(accum)/100.0

print 'Percent packets received:', percent,'%'

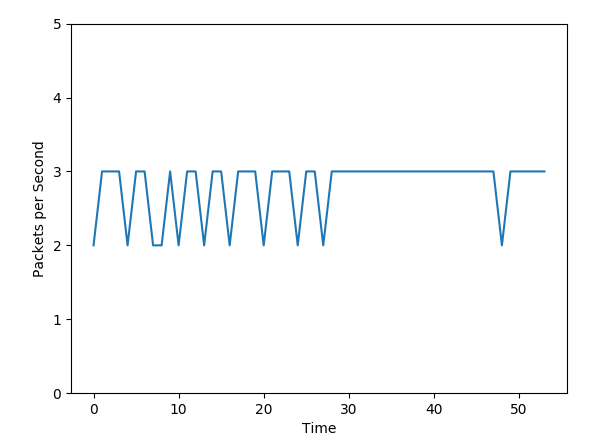
print "Average backoff time = " + str(average)

#raw\_input("Press Enter to read next file")

Question 10)

****

Question 11)



Question 12)

The throughput from Experiment 5.4 is between 2 and 3 packets per second. However, the common throughput is 3 packets per second. In Experiment 5.5, the throughput fluctuates between 2 and 3 packets per second at first and then it varies between 3 and 4 packets per second. Finally, in Experiment 5.6, the throughput varies between 2 and 3 packets per second and then it remains constant at 3 packets per second with very little variation. It can be concluded that in order to have an optimal throughput when there are multiple nodes it is necessary to have a large backoff exponent. If multiple nodes and a small backoff exponent (Experiment 5.4 where the backoff exponent is equal to 2 and the CCA threshold is equal to -95 dBm), then there are going to be more collisions and the throughput is going to get reduced. Moreover if there are many nodes and a very high CCA (Experiment 5.6 where the backoff exponent is equal to 2 and the CCA threshold is equal to 0 dBm), the throughput is going to get reduced even more because more signal strength is needed to be able to transfer the packets. In contrast, if there are many nodes, a large backoff exponent, equal to 5, and the CCA threshold is low, -95 dBm, then the throughput in theory should increase compared to the previous two results. Our experimental data yields to the same conclusion because at its maximum transfer rate it reached 4 packets per second.