Introduction to Wireless and Mobile Networking — Homework 4

Student ID: b03901032　Name: Jason Kuo 郭子生
Github: https://github.com/awinder0230/2017-Spring-Wireless-and-Mobile-Networking

Problem Description:

19 base stations are located in an urban area with temperature 27, which form a 19-cell map shown in Fig. 1. The coordination of the central BS is (0, 0) and ISD (inter site distance) is 500 m. The channel bandwidth is 10MHz. All BSs use the same carrier frequency (frequency reuse factor =1).The power of each base station is 33dBm. The power of each mobile device is 0dBm. The transmitter antenna gain and the receiver antenna gain for each device, including base station and mobile devices, are 14 dB. The height of each base station is 1.5m, which is located on the top of a 50m high building. The position of each mobile device is 1.5m high from the ground. Consider the path loss only radio propagation (without shadowing and fading). Use Two-ray-ground model as the propagation model for your simulation.

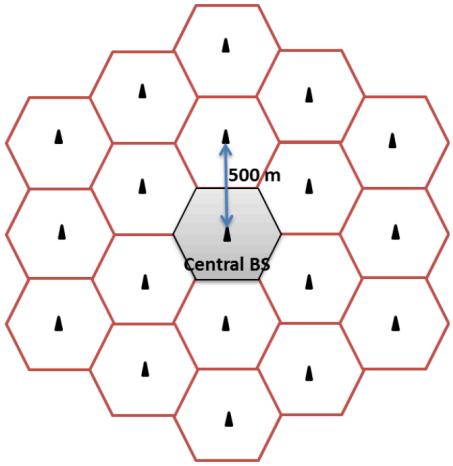

Figure 1.　19-cell map

Submission Files:

- report: b03901032_hw4_report.pdf
- readme: b03901032_hw4_readme.pdf
- codes:
  - main.m
  - gen_hexagon.m: a function to generate hexagon with radius r center at (x,y).
  - gen_hexrand.m: a function given hexagon, generate random dots inside the hexagon.
  - received_power_dB.m: given BS power, BS height, MS height, distance, transmitter gain, and receiver gain, compute received power in dB based on two-ray-ground model.
  - SINR.m: a function a function given signal, interference, and noise power, calculate SINR.
  - thermal_noise_power.m: a function given temperature and bandwidth, calculate thermal noise power.
  - two_ray_ground_model.m: given distance, height of transmitter and receiver, calculate gain of channel.
  - dB_2_watt.m: a function convert units from dB to Watt.
  - dBm_2_watt.m: a function convert units from dBm to Watt.
  - watt_2_dB.m: a function convert units from Watt to dB.

Usage:

1. Put all the *.m codes under the same directory.
2. Open Matlab and run main.m to get the simulation result.

Result:

1-1

Q: Please plot the location of the central BS and 50 uniformly random distributed mobile devices in the central cell. Don't plot the location of other BSs and other mobile devices in other cells. The unit of x-axis and y-axis should be "meter". The central BS is located at (0, 0). Also, use mark or color to differentiate the central BS from mobile devices.

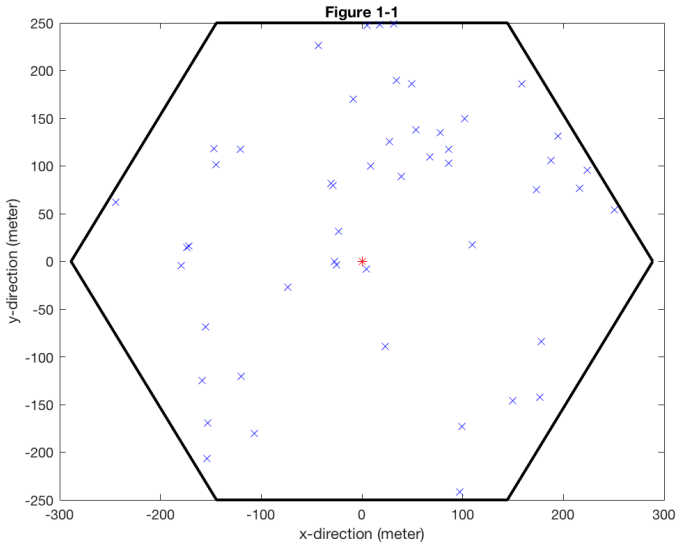

Figure. 1-1

Red * represents base station and blue x(s) represent mobile devices. The hexagon cell is generated by function gen_hexagon, and uniformly random distributed mobile devices are generated by function gen_hexrand.

1-2

Q: Based on the map in 1-1, please plot a figure with Shannon Capacity (bits/s) of a mobile device in a central BS as y-axis, and with the distance between the corresponding mobile device and the central BS as x-axis. Also, write down how to calculate the Shannon capacity of the mobile device in the central cell in your report.

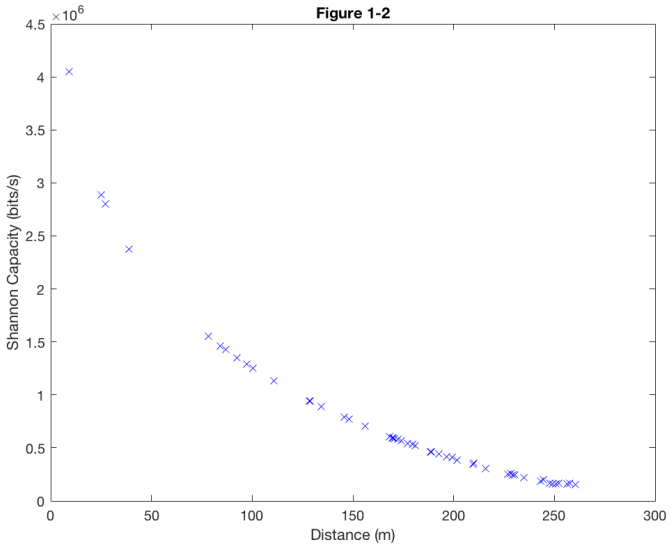

Figure. 1-2 Shannon Capacity = Bandwidth × $\log_2$ (1 + SINR)

Since total bandwidth is equally divided for the unicast communication of the 50 mobile devices, bandwidth for each mobile device equals (total bandwidth) / (number of mobile devices). S in SINR is the downlink power received from central base station, I in SINR is the downlink power received from all the other 18 base stations, and N in SINR is computed base on thermal noise model.

1-3

Q: Based on 1-1 and 1-2, design the CBR parameters {Xl, Xm, Xh} by yourself so that you can get different values of packet loss rate in the total simulation duration for each CBR parameter. Write down the CBR parameters you used in your simulation. Plot a histogram figure with the bits loss probability in the central BS as the y-axis and traffic load as the x-axis.

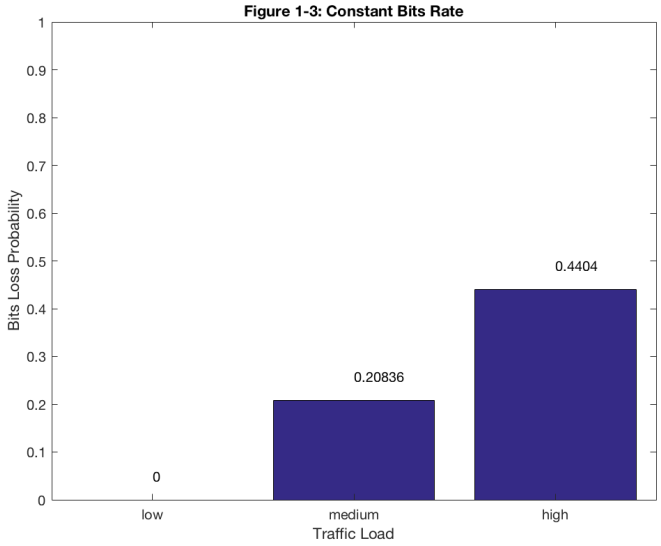

Figure. 1-3 $\{ X_{low}, X_{medium}, X_{high} \} = \{ 0.1 , 0.5 , 1 \}$ M bits/s

For each time step in the iteration, there are several cases that we might need to take care of. The following is the pseudo code of my algorithm:

```
for t = 1:simulation_time % iterate over 1000 seconds
  for k = 1:mobile_device_num % iteration over 50 MSs

    Bits = ComputeTransmissionBits();
    TotalBits = TotalBits + Bits;

    %% case 1: No loss bits
    if Bits + BitsInBuffer(k) <= Capacity(k)
      BitsInBuffer(k) = 0; % clear buffer

    %% case 2: newly arriving bits are larger than Shannon capacity
    elseif Bits >= Capacity(k)

      % subcase 2-1: buffer is not full yet
      if BitsInBuffer(k) < BufferMaxCapacity
        overflow = Bits - Capacity(k); % definition of overflow

        % buffer is not full even after adding overflow
        if BitsInBuffer(k) + overflow <= BufferMaxCapacity
          BitsInBuffer(k) = BitsInBuffer(k) + overflow;

        % buffer will be full after adding overflow
        else
          LossBits = LossBits + BitsInBuffer(k) + overflow - BufferMaxCapacity;
          BitsInBuffer(k) = BufferMaxCapacity;
        end

      % subcase 2-2: buffer is full already
      else % buffer is already full
        LossBits = LossBits + (Bits - Capacity(k));
      end

    %% case 3: newly arriving bits are less than Shannon capacity
    else
      BitsInBuffer(k) = BitsInBuffer(k) - (Capacity(k) - Bits); % relief buffer
    end

  end
end
```

During simulation, we need to iterate over all the mobile devices. For each mobile device, there are 3 main cases that we need to take care of, which are well commented in the pseudo codes above. Also, there may be lots of sub-cases in case 2 that may need to be aware of, for more details please refer to the pseudo codes above. To complete our simulation, simply iterate the algorithm over 3 types of constant bit rate. The subtle difference between different CBR is that the return value of function ComputeTransmissionBits( ) is different. For example, the return value for CBR=0.1 (Mbps) is 10^5, whereas the return value for CBR=1 (Mbps) is 10^6.

B-1

Q: Please plot the location of the central BS and 50 uniformly random distributed mobile devices in the central cell. Don't plot the location of other BSs and other mobile devices in other cells. The unit of x-axis and y-axis should be "meter". The central BS is located at (0, 0). Also, use mark or color to differentiate the central BS from mobile devices.

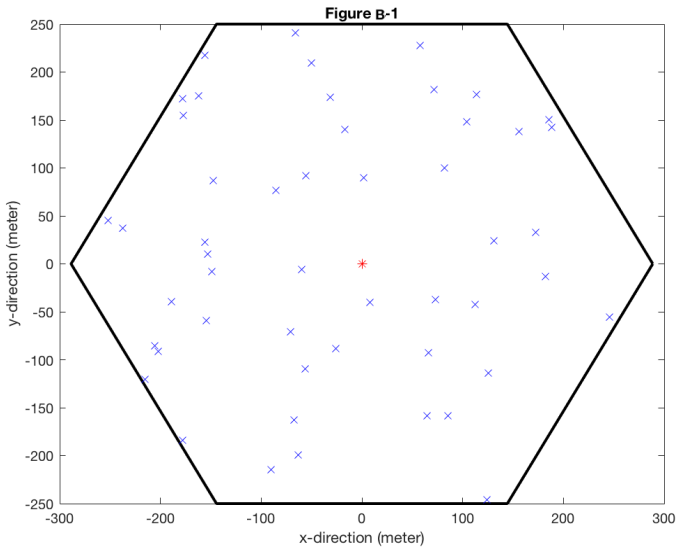

Figure. B-1

B-2

Q: Based on the map in B-1, please plot a figure with Shannon Capacity (bits/s) of a mobile device in a central BS as y-axis, and with the distance between the corresponding mobile device and the central BS as x-axis. Also, write down how to calculate the Shannon capacity of the mobile device in the central cell in your report.

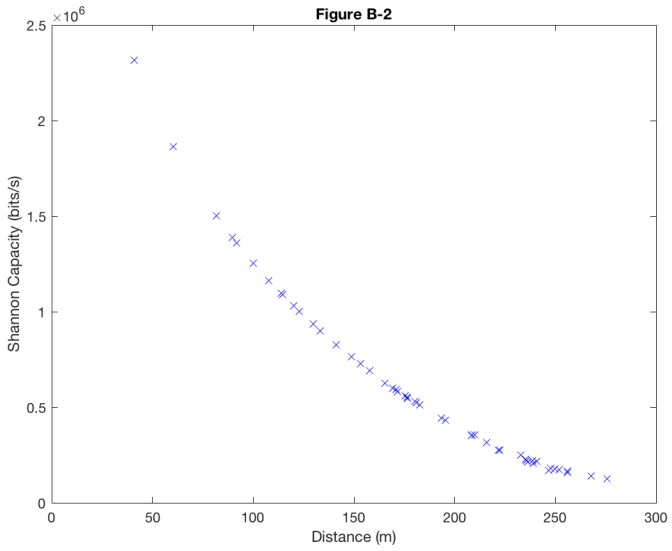

Figure. B-2 Shannon Capacity = Bandwidth × $\log_2$ (1 + SINR)

B-3

Q: Based on B-1 and B-2, design the Poisson traffic arrival parameters $\{\lambda_{low},\lambda_{medium},\lambda_{high}\}$ by yourself so that you can get different values of packet loss rate in the total simulation duration for each Poisson traffic arrival parameter. Write down the Poisson traffic arrival parameters you used in your simulation. Plot a histogram figure with the bits loss probability in the central BS as the y-axis and the traffic load as the x-axis.

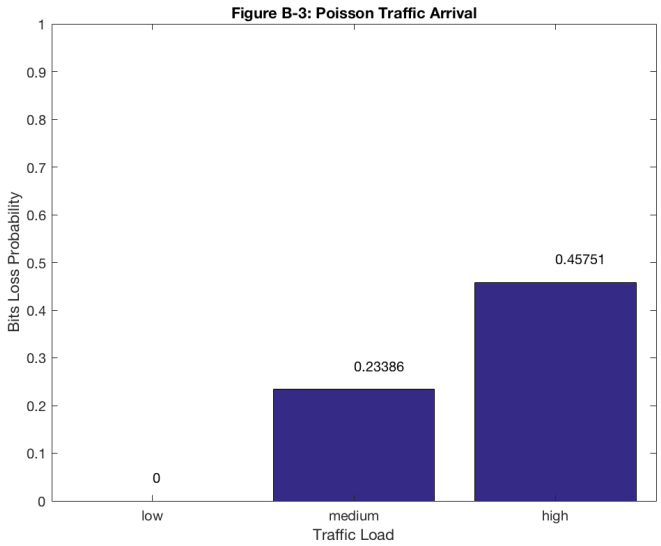

Figure. B-3 $\{\lambda_{low}, \lambda_{medium}, \lambda_{high}\} = \{ 0.1 , 0.5 , 1 \}$ M bits/s

Poisson distribution is a discrete probability distribution that gives the probability of the number of event occurrence during a fixed interval. The distribution is based on the assumption with known average event occurrence rate and the independency since last event.

$$P(k \text{ events in interval}) = e^{-\lambda}\frac{\lambda^k}{k!}$$

Here is a probability mass function for a Poisson distribution. lambda is the average number of event occurrence per time interval, and e is Euler's number. We then therefore being able to compute the "probability of observing k events during a time internal" according to the function above.

The algorithm for solving this problem is identical to the one described above, except the only difference resides in the return value of function ComputeTransmissionBits( ). Here the return value of function ComputeTransmissionBits( ) should be computed according to Poisson distribution given known lambda. Fortunately, there exist a function in Matlab named poissrnd() which returns Poisson random numbers given lambda as input. For example, calling the function poissrnd(2) ten times may get the result: 1, 0, 1, 2, 1, 3, 4, 2, 0, 0. By replacing function ComputeTransmissionBits with poissrnd, we are able to get the simulation results.

Last but not least, note that the simulation results in 1-3 and B-3 are quite similar, also note that we've assigned identical values to CBR and lambda. Recall that the definition of lambda is "the average number of event occurrence per time interval". Therefore, if the number of observed event for each time interval is exactly equal to lambda, this will become the circumstance in CBR.