Advanced Computer Networks (CS G525)
First Semester 2018-2019
Lab Session #3
Topic: Wireless Topology in *ns-3*

Objectives:

- **To get familiar with Wireless Topology in ns3.**

Running the First Wireless Topology file and Understanding it.

TASK:

Running the Wireless topology example and understanding various components.

```
$cd NS3repo/ns-3allinone/ns-3-dev/examples/wireless/wifi-tcp.cc
```

copy the wifi-tcp.cc file to the scratch folder and run the file

```
./waf –run wifi-tcp.cc
```

In this example, an HT station sends TCP packets to the access point. We report the total throughput received during a windw of 100 ms. The user can specify the application data rate and chose the variant of TCP i.e. congestion control algorithm to use

Creating the First Wireless Topology and Understanding it.

We Will Follow example **third.cc** to create the topology.

1. Network Topology Consist of :

   --Wireless nodes/links:
   - 3 STA nodes
   - 1 AP node
   - 802.11 links , non QoS mode , beaconing enabled

   --Wired node/links:

   - 2 nodes connected via PPP link
   - 4 nodes on CSMA LAN

2. Application:

   - Server on CSMA sub network , client on STA node.

3. NodeContainer Class, Create method:

    - NodeContainer wifiStaNodes
    - wifiStaNodes.Create (nWifi)
    - NodeContainer wifiApNode =p2pNodes.Get (0)

4. Set WifiChannel and WifiPhy

    - YansWifiChannelHelper channel = YansWifiChannelHelper :: Default()
    - YansWifiPhyHelper phy = YansWifiPhyHelper :: Default()

5. Associate Channel and Phy

    - Phy.SetChannel (channel.Create ());

6. Configure MAC Layer

    - WifiHelper wifi = WifiHelper :: Default();
    - wifi.SetRemoteStationManager ("ns3::AarfWifiManager");
    - mac.SetType ("ns3::StaWifiMac","Ssid", SsidValue (ssid),"ActiveProbing", BooleanValue (false));

7. Install MAC layer Properties to the devices

    - NetDeviceContainer staDevices;
    - staDevices = wifi.Install (phy, mac, wifiStaNodes);

8. Install Mobility models to nodes.

    - mobility.SetMobilityModel ("ns3::RandomWalk2dMobilityModel", "Bounds", RectangleValue (Rectangle (-50, 50, -50, 50)));
    - mobility.Install (wifiStaNodes);

We can use “ns3::ConstantPositionMobilityModel” in SetMobilityModel also.

For more information: https://www.nsnam.org/docs/release/3.8/tutorial/tutorial_27.html

Simulating The Topology:

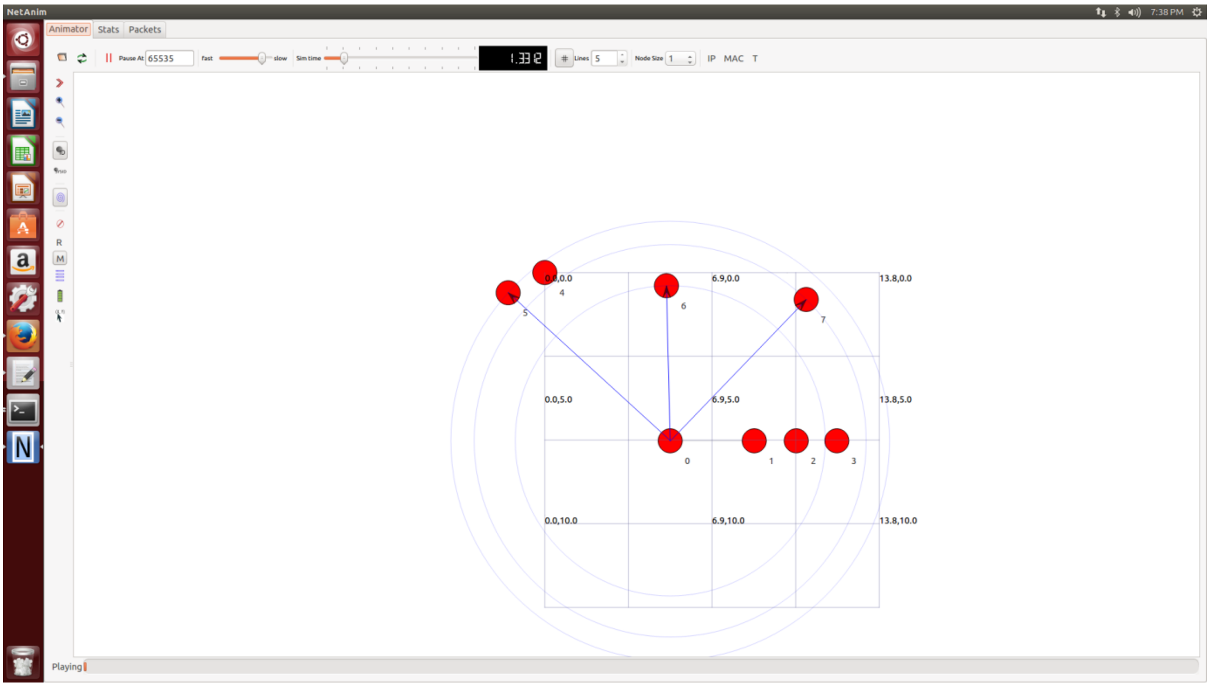
Add the following lines to add the animation interface to the topology file.

```
AnimationInterface anim(animFile);
   Ptr<Node> n = p2pNodes.Get(0);
   anim.SetConstantPosition(n, 5.45, 5);
   n = p2pNodes.Get(1);
   anim.SetConstantPosition(n, 7.99, 5);
   n = csmaNodes.Get(0);
```

```
anim.SetConstantPosition(n, 2, 4);
n = csmaNodes.Get(1);
anim.SetConstantPosition(n, 8, 2);
n = csmaNodes.Get(2);
anim.SetConstantPosition(n, 8, 6);
n = wifiStaNodes.Get(0);
anim.SetConstantPosition(n, 10, 4.0);
    n = wifiStaNodes.Get(1);
anim.SetConstantPosition(n, 12, 4.0);
   n = wifiStaNodes.Get(2);
anim.SetConstantPosition(n, 14, 4.0);
```

We can change the x and y co-ordinate according to the user preferences

Run the File To see the output which will look like below(the output may change from system to system).



TASK:

Make the Following Changes and try to achieve the output.

- Only 4 STA nodes and 1 AP node.
  --Infastructure mode
  -- no wired links → this means no CSMA,no PPP links.
- Non-Qos Mode
- Enable ASCII and PCAP tracing on all interfaces.
- Rest of the attribute values: use from the examples like mobility models.
- Application:
  --From STA 2 to STA 4

--UdpEchoServer on port 70.

TASK(optional):

Use WifiMax examples from the folder:

$cd NS3repo/ns-3allinone/ns-3-dev/src/wimax/examples

Run the files and try to see the output and understand the concept of using WifiMax and try to generate the topology with

- 1 Base Station
- 4 Subscriber Stations

WireShark:

Installation:

sudo apt-get install wireshark

The script has following lines to generate .pcap traces to be loaded in wireshark.

```
if (tracing == true)
   {
     pointToPoint.EnablePcapAll ("third");
     phy.EnablePcap ("third", apDevices.Get (0));
     csma.EnablePcap ("third", csmaDevices.Get (0), true);
   }
```

This causes the script to generate .pcap file for each interface. Open the Wireshark by typing **sudo wireshark** on the terminal. Just open trace by *file->open-> <select file>*. Now you can see all of the packets that went through the respective interface.

Throughput Calculation:

For Analysing the Throughput of the particular flow you will need to add the following line:
Before main function

```
void
CalculateThroughput ()
{
  Time now = Simulator::Now ();                                   /* Return the simulator's virtual time. */
  double cur = (sink->GetTotalRx () - lastTotalRx) * (double) 8 / 1e5;     /* Convert Application RX
Packets to MBits. */
  std::cout << now.GetSeconds () << "s: \t" << cur << " Mbit/s" << std::endl;
  lastTotalRx = sink->GetTotalRx ();
  Simulator::Schedule (MilliSeconds (100), &CalculateThroughput);
}
```

Add the variables above the function

```
Ptr<PacketSink> sink;                    /* Pointer to the packet sink application */
uint64_t lastTotalRx = 0;
```

You will need to add the variables such as data rate and payload size also.
After starting simuation you will need to add following lines:

```
double averageThroughput = ((sink->GetTotalRx () * 8) / (1e6 * simulationTime));

  Simulator::Destroy ();

  if (averageThroughput < 50)
    {
      NS_LOG_ERROR ("Obtained throughput is not in the expected boundaries!");
      exit (1);
    }
  std::cout << "\nAverage throughput: " << averageThroughput << " Mbit/s" << std::endl;
```

Above part will give you the throughput for one flows.

Exercises:

1.Make the following changes to third.cc file

- Make the Same connection with TCP receiver and Sink instead of UDP application server and observe the throughput
- Make the Topology with both UDP Application server and TCP source and sink and observe the output
- Also make throughput calculation for every flows.

Note:To make TCP sink and source you can take help from the wifi-tcp.cc file.

2.Download Topology.cc file which was used earlier in the Lab and make the following changes:

- Make the left part of the dumbbell topology as Wifi Station nodes
- Make one of the nodes in the bottleneck link as Access Point.
- Make the Station nodes as Mobile nodes
- Observe the output.