Network Simulation

A Simulation Scenario

Group 6
BI11-222 Kiều Tuấn Phương
BI11-180 Nguyễn Đức Minh
BI11-156 Nguyễn Bảo Long
BI11-035 Lê Đức Bách
BI11-239 Nguyễn Xuân Thái

Problem:
Consider CSMA/CA protocol in Wifi networks that working in ad-hoc mode. Evaluate performance of the protocol without RTS/CTS scheme when the number of nodes within a communication range increases from 2 to 30.
Hints:
Please follow steps to create a network simulation

1. Design
2. Implementing
3. Data Collection and Report

Task analysis

1. Design Network Topology

- [ ] **Disable RTS/CTS:**

```
UintegerValue threshold = 1000; // original value:500
Config::SetDefault("ns3::WifiRemoteStationManager::RtsCtsThreshold", threshold);
```

- [ ] **Create Nodes & install Wifi model:**

```
  // Create the nodes that compose the network
  NodeContainer nodes;
  nodes.Create(nNodes);

  // Configure the PHY and channel helpers
  YansWifiChannelHelper channel = YansWifiChannelHelper::Default();
  YansWifiPhyHelper phy;
  /* Create a channel object and associate it to PHY layer  object manager to make
sure that
    all the PHY layer objects created by the YansWifiPhyHelper share the same
underlying channel*/
  phy.SetChannel(channel.Create());

  // WifiMacHelper is used to set MAC parameters.
  WifiMacHelper mac;

  /* Instantiate WifiHelper (By default, configure the standard in use to be
802.11ax
    and configure a compatible rate control algorithm - IdealWifiManager)*/
  WifiHelper wifi;

  // Configure the MAC layer
  mac.SetType("ns3::AdhocWifiMac");

  NetDeviceContainer devices;
  devices = wifi.Install(phy, mac, nodes);
```

☐ **Install mobility model:**

```
/* Instantiate a MobilityHelper object and set some attributes
controlling
the "position allocator" functionality */
MobilityHelper mobility;
mobility.SetPositionAllocator(
"ns3::GridPositionAllocator",
"MinX", DoubleValue(0.0),
"MinY", DoubleValue(0.0),
"DeltaX", DoubleValue(5.0),
"DeltaY", DoubleValue(5.0),
"GridWidth", UintegerValue(5),
"LayoutType", StringValue("RowFirst"));
/* Set the mobility model to be ns3::ConstantPositionMoblityModel to
fixed the position of the devices */
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.Install(nodes);
```

☐ **Install Internet Stack:**

```
// Use InternetStackHelper to install protocol stacks
InternetStackHelper stack;
stack.Install(nodes);
// Use Ipv4AddressHelper to assign IP addresses to our device interfaces
Ipv4AddressHelper address;
// Use the network 10.1.1.0 to create the addresses needed for our
devices
address.SetBase("10.1.1.0", "255.255.255.0");
/* Save the created interfaces in a container to make it easy to pull
out
addressing information later for use in setting up the applications */
Ipv4InterfaceContainer nodeInterfaces;
nodeInterfaces = address.Assign(devices);
```

2. Setup Applications

☐ **UdpEchoServer:**

```
// Create a UdpEchoServerHelper and provide the server port number
UdpEchoServerHelper echoServer(9);
// Instantiate the server on the choosen server node
ApplicationContainer serverApps =
echoServer.Install(nodes.Get(serverNode));
serverApps.Start(Seconds(2.0));
serverApps.Stop(Seconds(simTime));
```

☐ **UdpEchoClient:**

```
// Create a UdpEchoClientHelper and provide the remote address and port
UdpEchoClientHelper echoClient(nodeInterfaces.GetAddress(serverNode),
9);
echoClient.SetAttribute("MaxPackets", UintegerValue(maxPackets)); //
Default is 10
echoClient.SetAttribute("Interval", TimeValue(Seconds(interval)));//
Default is 1s
echoClient.SetAttribute("PacketSize", UintegerValue(packetSize)); //
Default is 512B
// Install the client on every other node except the serverNode
for (uint32_t i = 0; i < nNodes; i++) {
if (i == serverNode) continue; // Exclude the server node
ApplicationContainer clientApp = echoClient.Install(nodes.Get(i));
clientApp.Start(Seconds(2.0));
clientApp.Stop(Seconds(simTime));
}
```

3. Collect Data

☐ **Enable pcap:**

```
// Enable pcap
if (pcap) {
    phy.EnablePcap("final", devices.Get(serverNode), true);
}
```

☐ **Flow Monitor**

```
// Install flowMonitor to collect data
Ptr<FlowMonitor> flowMonitor;
FlowMonitorHelper flowHelper;
flowMonitor = flowHelper.InstallAll();
```

```
// Save the result of flow monitor in an xml file
char filename[100];
sprintf(filename, "final-%d-nodes.xml", nNodes);
flowMonitor->SerializeToXmlFile(filename, true, true);
```

☐ **Collect Data for Visualization**

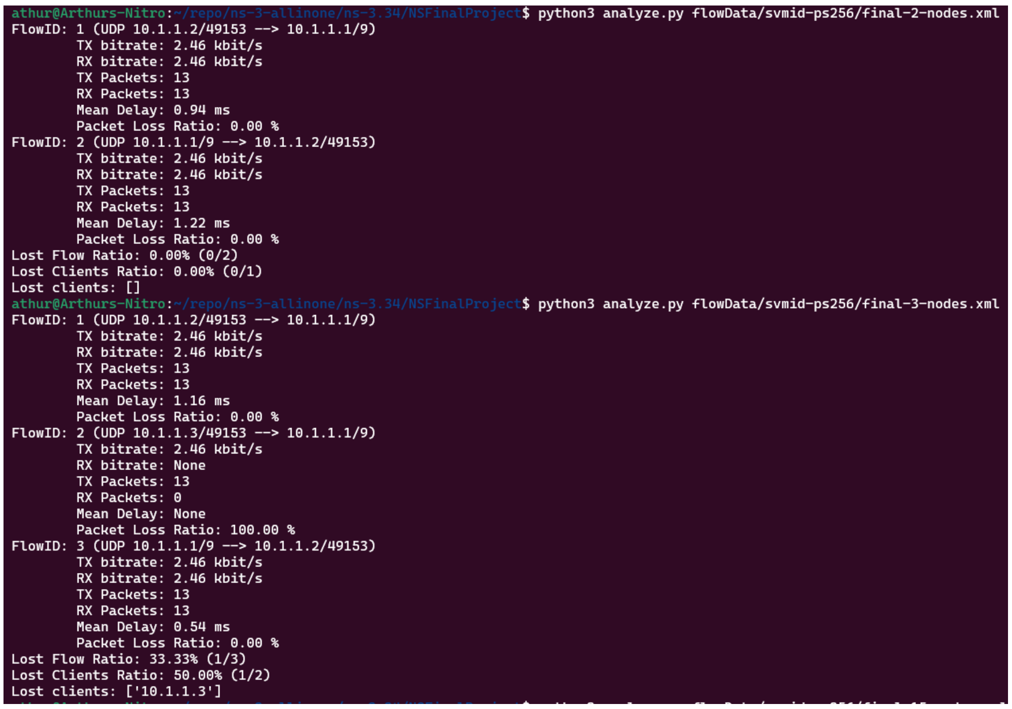
```
// Use netanim to visualize the experiment and save the result in an xml
```

```
file
char animfile[100];
sprintf(animfile, "anim-%d-nodes.xml", nNodes);
AnimationInterface anim(animfile);
```

4. Analyze Data

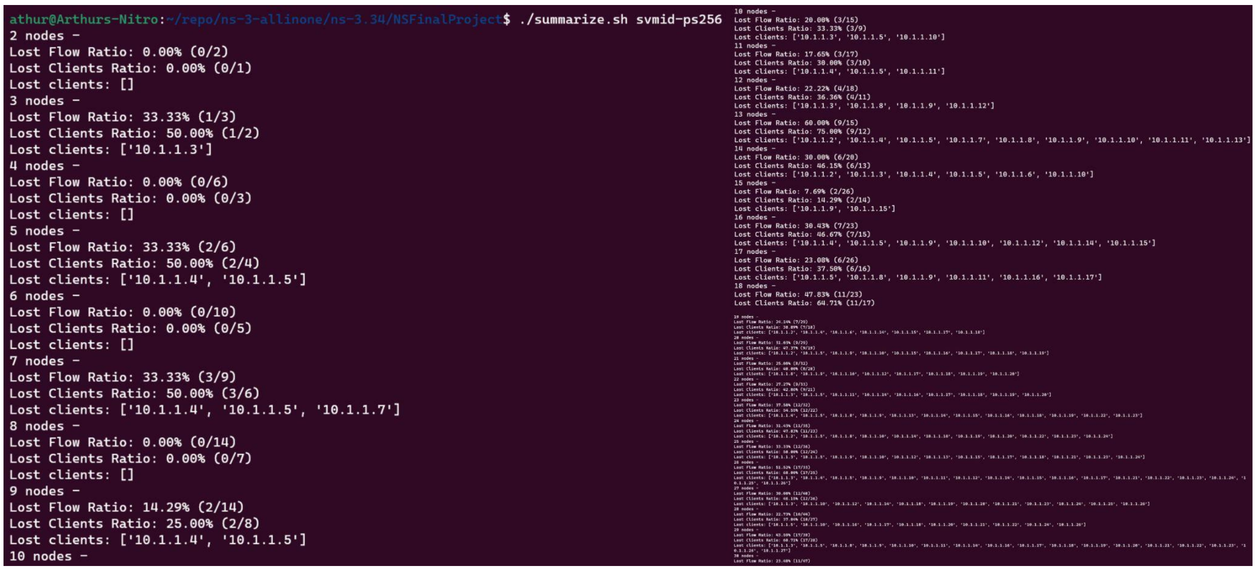
Data from the flows contains a lot of interesting information such as

- The time the first and last packet is transmitted and received
- The total delay
- The total bytes and packets transmitted and received
- Number of lost packets



```
athur@Arthurs-Nitro:~/repo/ns-3-allinone/ns-3.34/NSFinalProject$ python3 analyze.py flowData/svmid-ps256/final-2-nodes.xml
FlowID: 1 (UDP 10.1.1.2/49153 --> 10.1.1.1/9)
        TX bitrate: 2.46 kbit/s
        RX bitrate: 2.46 kbit/s
        TX Packets: 13
        RX Packets: 13
        Mean Delay: 0.94 ms
        Packet Loss Ratio: 0.00 %
FlowID: 2 (UDP 10.1.1.1/9 --> 10.1.1.2/49153)
        TX bitrate: 2.46 kbit/s
        RX bitrate: 2.46 kbit/s
        TX Packets: 13
        RX Packets: 13
        Mean Delay: 1.22 ms
        Packet Loss Ratio: 0.00 %
Lost Flow Ratio: 0.00% (0/2)
Lost Clients Ratio: 0.00% (0/1)
Lost clients: []
athur@Arthurs-Nitro:~/repo/ns-3-allinone/ns-3.34/NSFinalProject$ python3 analyze.py flowData/svmid-ps256/final-3-nodes.xml
FlowID: 1 (UDP 10.1.1.2/49153 --> 10.1.1.1/9)
        TX bitrate: 2.46 kbit/s
        RX bitrate: 2.46 kbit/s
        TX Packets: 13
        RX Packets: 13
        Mean Delay: 1.16 ms
        Packet Loss Ratio: 0.00 %
FlowID: 2 (UDP 10.1.1.3/49153 --> 10.1.1.1/9)
        TX bitrate: 2.46 kbit/s
        RX bitrate: None
        TX Packets: 13
        RX Packets: 0
        Mean Delay: None
        Packet Loss Ratio: 100.00 %
FlowID: 3 (UDP 10.1.1.1/9 --> 10.1.1.2/49153)
        TX bitrate: 2.46 kbit/s
        RX bitrate: 2.46 kbit/s
        TX Packets: 13
        RX Packets: 13
        Mean Delay: 0.54 ms
        Packet Loss Ratio: 0.00 %
Lost Flow Ratio: 33.33% (1/3)
Lost Clients Ratio: 50.00% (1/2)
Lost clients: ['10.1.1.3']
```

Summarize all the analyzes:

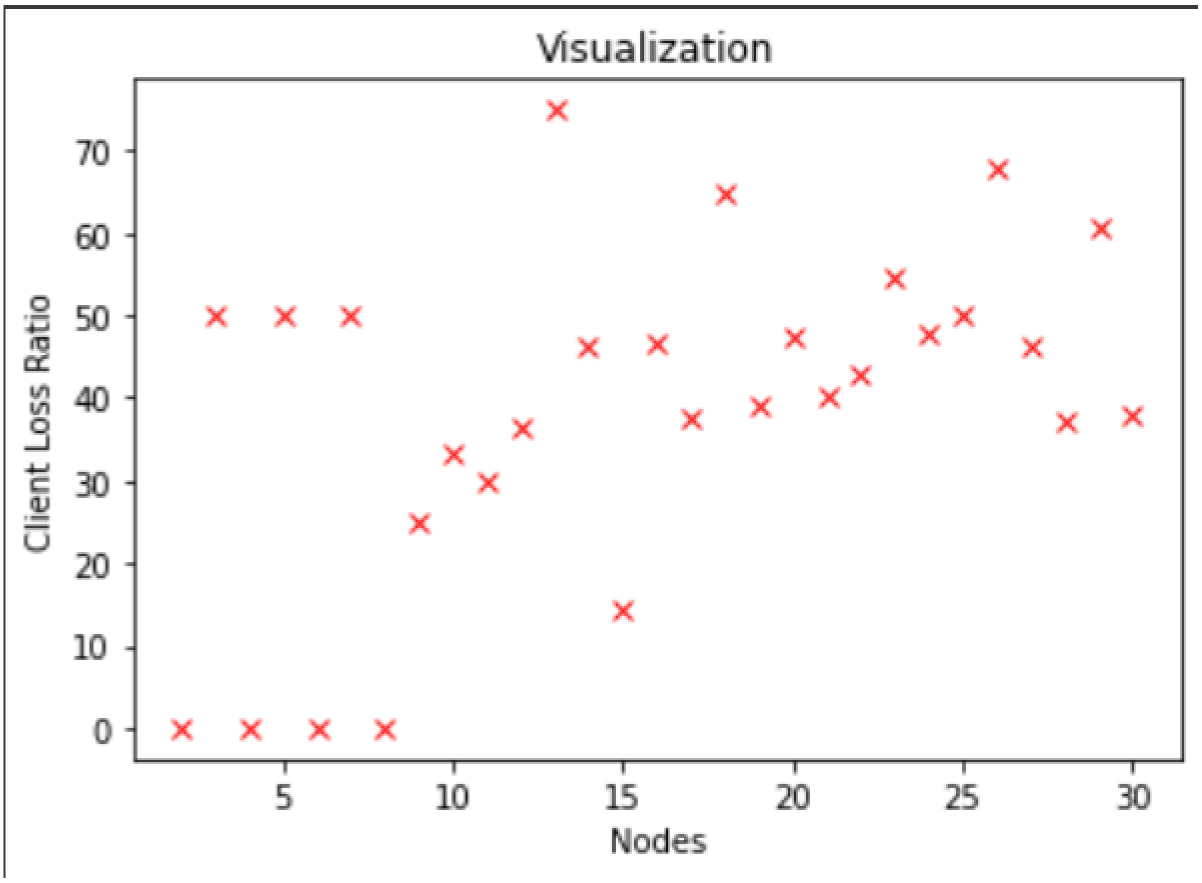


```
athur@Arthurs-Nitro:~/repo/ns-3-allinone/ns-3.34/NSFinalProject$ ./summarize.sh svmid-ps256
2 nodes -
Lost Flow Ratio: 0.00% (0/2)
Lost Clients Ratio: 0.00% (0/1)
Lost clients: []
3 nodes -
Lost Flow Ratio: 33.33% (1/3)
Lost Clients Ratio: 50.00% (1/2)
Lost clients: ['10.1.1.3']
4 nodes -
Lost Flow Ratio: 0.00% (0/6)
Lost Clients Ratio: 0.00% (0/3)
Lost clients: []
5 nodes -
Lost Flow Ratio: 33.33% (2/6)
Lost Clients Ratio: 50.00% (2/4)
Lost clients: ['10.1.1.4', '10.1.1.5']
6 nodes -
Lost Flow Ratio: 0.00% (0/10)
Lost Clients Ratio: 0.00% (0/5)
Lost clients: []
7 nodes -
Lost Flow Ratio: 33.33% (3/9)
Lost Clients Ratio: 50.00% (3/6)
Lost clients: ['10.1.1.4', '10.1.1.5', '10.1.1.7']
8 nodes -
Lost Flow Ratio: 0.00% (0/14)
Lost Clients Ratio: 0.00% (0/7)
Lost clients: []
9 nodes -
Lost Flow Ratio: 14.29% (2/14)
Lost Clients Ratio: 25.00% (2/8)
Lost clients: ['10.1.1.4', '10.1.1.5']
10 nodes -
```

```
10 nodes -
Lost Flow Ratio: 20.00% (3/15)
Lost Clients Ratio: 33.33% (3/9)
Lost clients: ['10.1.1.3', '10.1.1.5', '10.1.1.10']
11 nodes -
Lost Flow Ratio: 17.65% (3/17)
Lost Clients Ratio: 30.00% (3/10)
Lost clients: ['10.1.1.4', '10.1.1.5', '10.1.1.11']
12 nodes -
Lost Flow Ratio: 22.22% (4/18)
Lost Clients Ratio: 36.36% (4/11)
Lost clients: ['10.1.1.3', '10.1.1.8', '10.1.1.9', '10.1.1.12']
13 nodes -
Lost Flow Ratio: 60.00% (9/15)
Lost Clients Ratio: 75.00% (9/12)
Lost clients: ['10.1.1.2', '10.1.1.4', '10.1.1.5', '10.1.1.7', '10.1.1.8', '10.1.1.9', '10.1.1.10', '10.1.1.11', '10.1.1.13']
14 nodes -
Lost Flow Ratio: 30.00% (6/20)
Lost Clients Ratio: 46.15% (6/13)
Lost clients: ['10.1.1.2', '10.1.1.3', '10.1.1.4', '10.1.1.5', '10.1.1.6', '10.1.1.10']
15 nodes -
Lost Flow Ratio: 7.69% (2/26)
Lost Clients Ratio: 14.29% (2/14)
Lost clients: ['10.1.1.9', '10.1.1.15']
16 nodes -
Lost Flow Ratio: 30.43% (7/23)
Lost Clients Ratio: 46.67% (7/15)
Lost clients: ['10.1.1.4', '10.1.1.5', '10.1.1.9', '10.1.1.10', '10.1.1.12', '10.1.1.14', '10.1.1.15']
17 nodes -
Lost Flow Ratio: 23.08% (6/26)
Lost Clients Ratio: 37.50% (6/16)
Lost clients: ['10.1.1.5', '10.1.1.8', '10.1.1.9', '10.1.1.11', '10.1.1.16', '10.1.1.17']
18 nodes -
Lost Flow Ratio: 47.83% (11/23)
Lost Clients Ratio: 64.71% (11/17)
```

```
19 nodes -
Lost Flow Ratio: 24.14% (7/29)
Lost Clients Ratio: 38.89% (7/18)
Lost clients: ['10.1.1.2', '10.1.1.4', '10.1.1.6', '10.1.1.14', '10.1.1.15', '10.1.1.17', '10.1.1.18']
20 nodes -
Lost Flow Ratio: 31.03% (9/29)
Lost Clients Ratio: 47.37% (9/19)
Lost clients: ['10.1.1.2', '10.1.1.5', '10.1.1.9', '10.1.1.10', '10.1.1.15', '10.1.1.16', '10.1.1.17', '10.1.1.18', '10.1.1.19']
21 nodes -
Lost Flow Ratio: 25.00% (8/32)
Lost Clients Ratio: 40.00% (8/20)
Lost clients: ['10.1.1.8', '10.1.1.9', '10.1.1.10', '10.1.1.12', '10.1.1.17', '10.1.1.18', '10.1.1.19', '10.1.1.20']
22 nodes -
Lost Flow Ratio: 27.27% (9/33)
Lost Clients Ratio: 42.86% (9/21)
Lost clients: ['10.1.1.3', '10.1.1.5', '10.1.1.11', '10.1.1.14', '10.1.1.16', '10.1.1.17', '10.1.1.18', '10.1.1.19', '10.1.1.20']
23 nodes -
Lost Flow Ratio: 37.50% (12/32)
Lost Clients Ratio: 54.55% (12/22)
Lost clients: ['10.1.1.4', '10.1.1.5', '10.1.1.8', '10.1.1.9', '10.1.1.13', '10.1.1.14', '10.1.1.15', '10.1.1.16', '10.1.1.18', '10.1.1.19', '10.1.1.22', '10.1.1.23']
24 nodes -
Lost Flow Ratio: 31.43% (11/35)
Lost Clients Ratio: 47.83% (11/23)
Lost clients: ['10.1.1.2', '10.1.1.5', '10.1.1.8', '10.1.1.10', '10.1.1.14', '10.1.1.18', '10.1.1.19', '10.1.1.20', '10.1.1.22', '10.1.1.23', '10.1.1.24']
25 nodes -
Lost Flow Ratio: 33.33% (12/36)
Lost Clients Ratio: 50.00% (12/24)
Lost clients: ['10.1.1.3', '10.1.1.5', '10.1.1.9', '10.1.1.10', '10.1.1.12', '10.1.1.13', '10.1.1.15', '10.1.1.17', '10.1.1.18', '10.1.1.21', '10.1.1.23', '10.1.1.24']
26 nodes -
Lost Flow Ratio: 51.52% (17/33)
Lost Clients Ratio: 68.00% (17/25)
Lost clients: ['10.1.1.3', '10.1.1.4', '10.1.1.5', '10.1.1.9', '10.1.1.10', '10.1.1.11', '10.1.1.12', '10.1.1.14', '10.1.1.15', '10.1.1.16', '10.1.1.17', '10.1.1.21', '10.1.1.22', '10.1.1.23', '10.1.1.24', '1
0.1.1.25', '10.1.1.26']
27 nodes -
Lost Flow Ratio: 30.00% (12/40)
Lost Clients Ratio: 46.15% (12/26)
Lost clients: ['10.1.1.3', '10.1.1.10', '10.1.1.12', '10.1.1.14', '10.1.1.18', '10.1.1.19', '10.1.1.20', '10.1.1.21', '10.1.1.23', '10.1.1.24', '10.1.1.25', '10.1.1.26']
28 nodes -
Lost Flow Ratio: 22.73% (10/44)
Lost Clients Ratio: 37.04% (10/27)
Lost clients: ['10.1.1.5', '10.1.1.10', '10.1.1.16', '10.1.1.17', '10.1.1.18', '10.1.1.20', '10.1.1.21', '10.1.1.22', '10.1.1.24', '10.1.1.26']
29 nodes -
Lost Flow Ratio: 43.59% (17/39)
Lost Clients Ratio: 60.71% (17/28)
Lost clients: ['10.1.1.3', '10.1.1.5', '10.1.1.8', '10.1.1.9', '10.1.1.10', '10.1.1.11', '10.1.1.14', '10.1.1.16', '10.1.1.17', '10.1.1.18', '10.1.1.19', '10.1.1.20', '10.1.1.21', '10.1.1.22', '10.1.1.23', '1
0.1.1.26', '10.1.1.27']
30 nodes -
Lost Flow Ratio: 23.40% (11/47)
```

Use matplotlib to visualize the lost clients ratio:



⇒ *When the number of nodes increases, the ratio of lost client also increases*

5. Explain the Results

In ad-hoc wireless networks where all devices are connected to each other, if we disable RTS/CTS, data collision is more likely to happen. Therefore when the number of nodes is increased, the number of clients and the number of packets being transmitted is also increased. More collisions happen and more packets are lost.

6. Code Implementation

All the source code at:

https://github.com/KieuTuanPhuong/NSFinalProject