Performance analysis of single channel WLAN Systems in dense deployments

A Project Report

submitted by

DEEPAK PADAMATA

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF Electrical Engineering
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

May 2018

THESIS CERTIFICATE

This is to certify that the thesis titled **Throughput estimation for single channel WLAN Systems in dense deployment scenarios**, submitted by **Deepak Padamata**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. K Giridhar
Research Guide,
Professor,
Dept. of Electrical Engineering,
IIT-Madras, 600 036

Place: Chennai

Date: 21st May 2018

ACKNOWLEDGEMENTS

I would like to thank Prof. Giridhar for his constant guidance during my research. His emphasis on insights and intuition has been crucial in understanding various topics in wireless communications and for assimilating complex ideas. I had a wonderful opportunity to be a TA for Prof. Balaji Srinivasan. His dedication and his down-to-earth personality is very inspiring and my interactions with him have been extremely gratifying. I would also like to thank Prof. Venkatesh Ramaiyan whose help and guidance has been instrumental in the completion my project.

The five years of my stay at IITM have been my most formative years with amazing people with the most beautiful minds all around me. I would like to convey my heartfelt gratitude to my friends Swarup, Chicha and Phani and many others for making my stay here such an intense experience. I would also like to thank CR Venkatesh sir and my lab mates Madhan, Manideep, Parthiban, Koushik, Gokulram, Abhijith, Shweta, Sairaj. Shubhankar for all the discussions (technical, and otherwise).

Finally I would like to thank my parents and my sister for their unwavering belief in me and their love.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS **i**

LIST OF FIGURES **iv**

ABBREVIATIONS **v**

1 INTRODUCTION **1**

1.1 Need for Studying Dense Local Wireless Networks 1

1.2 Contributions of the Project 2

2 WiFi and 802.11n **3**

2.1 Introduction 3

2.2 802.11 Protocol Family 3

2.3 802.11n 4

2.3.1 OFDM PHY 4

2.3.2 MAC 5

2.3.3 Summary - improvements made in 802.11n 6

3 Network Simulator - 3 **7**

3.1 Introduction 7

3.2 Simulation flow 7

3.3 Internal simulation system of NS-3 8

4 Simulation parameters **9**

4.1 Node Definition 9

4.2 PHY and channel definition 9

4.3 Physical layout 12

4.4 SSID and IP address allocation 13

4.5 Applications 14

4.6 Data Processing and Packet Capture 15

5 Simulation Results **16**

6 Conclusions and Future Work **20**

7 Other Work - Rayleigh Fading Channel Generator Using the Dent Model **21**

LIST OF FIGURES

2.1 Subcarrier structure in 802.11n 5

4.1 Node Positioning 12

5.1 UDP - Downlink Only 17

5.2 UDP - Uplink and Downlink 17

5.3 TCP - Downlink 18

5.4 TCP - Uplink and Downlink 18

5.5 Best Transmit power scenario comparisons 19

ABBREVIATIONS

OFDM	Orthogonal Frequency Division Multiplexing
LTE	Long Term Evolution
GSM	Global System for Mobile communication
WLAN	Wireless Local Area Network
NS-3	Network Simulator, Version 3
WiFi	Wireless Fidelity
DSSS	Direct-Sequence Spread Spectrum
FHSS	Frequency-Hopping Spread Spectrum
MIMO	Multiple-Input Multiple-Output
PHY	Physical - a layer of the OSI model
MAC	Medium Access Control - a layer of the OSI model
OSI	Open Systems Interconnection
HT	High=Throughput
FFT	Fast Fourier Transform
QAM	Quadrature Amplitude Modulation
LDPC	Low Density Parity Check
IP	Internet Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
MCS	Modulation and Coding Scheme
AP	Access Point
STA	Station
YANS	Yet Another Network Simulator
RTS/CTS	Ready To Send / Clear To Send
SSID	Service Set Identifier

CHAPTER 1

INTRODUCTION

Wireless communication is one of the most fundamental form of communications - speaking and seeing being wireless, and this was practiced in human society from a time way before the introduction of wire-line communication. However, long distance wireless communications started with Nikola Tesla, based on the works of Maxwell and Hertz on Electro-Magnetic (EM) waves, and demonstrated the transmission of information over them. Marconi, however , was awarded a Nobel Prize for Physics in 1909 after his famous demonstration of wireless communication from a boat to Isle of Wight in the English Channel in the year 1898 [(3)]. But even before Marconi demonstrated wireless communication, an Indian scientist J.C. Bose had held a patent for transmitting information over EM waves [(2)]. Since then, wireless communication has come a long way with many advancements and improvements. Introductions of digital transmission standards, the latest being 4G LTE and 5G coming up in the near future. Along with the standards, the design of data carriers has also evolved with the initial Global System for Mobile Communication (GSM) standard operating on Time Division Multiplexing (TDM), Code Division Multiple Access (CDMA), and Frequency Division Multiplexing (FDM), to now in 4G with data being carried over OFDM. With ever increasing services available on the mobile devices propelled by consumer market demand, the latest mobile technology needs to operate over a wide bandwidth and handle wide range of channel conditions. Since OFDM is able to meet these demands, it has become one of the most popular communication techniques. The same OFDM standard is also used in another common technology, which is Wireless Fidelity (WiFi). WiFi is a Wireless Local Area Network (WLAN) standard, the latest widely-used standards being 802.11ac and 802.11n.

1.1 Need for Studying Dense Local Wireless Networks

It has been well established that the demand for wireless connectivity is rising rapidly with the advent of Internet of Things (IoT) and mobile users are demanding higher data rates than ever with HD video and fast internet becoming ubiquitous. Traditional deployments for 4G and 5G need to be supplemented with other solutions to enhance the browsing experience for users and provide better services. Many users now prefer to consume data wirelessly even in their homes, where they might have broadabnd simply because it is the simpler way. In response to this providers have looked into offloading traffic from the cellular network to a local network in order serve these data-hungry users. WiFi has been proposed as a potential solution to this problem, due to it's local nature and ease of deployment.

1.2 Contributions of the Project

In this work, we review a WiFi system using the 802.11n protocol, in the context of regular deployment over a geographical area, with a high density of users. We use the **NS-3** simulator for network simulation. We use different traffic types and internet protocols, and observe the behavior of the network, it's sum throughput and trends as we increase the number of Access Points(APs) on the channel and users connected per AP in a small geographical area.

The thesis is organized as follows

Chapter 2 discusses the 802.11n standard and WiFi in general. It illustrates some features of the IEEE 802.11 protocol, particularly focusing on 802.11n, since it is the protocol of choice for our simulations.

Chapter 3 introduces NS-3, which is a widely used network simulation tool and we also go through it's inner workings and how a network is simulated in it. NetAnim can be used to display visually the setup and tcpdump is used to examine packet data captured over the network.

Chapter 4 deals with the simulation setup and explains our chosen parameters and illustrates what parameters can be used to change network conditions as per requirements.

Chapter 5 gives conclusions and some directions for future research.

CHAPTER 2

WiFi and 802.11n

2.1 Introduction

Wireless networks are popular and needed for various reasons, ranging from being inexpensive due to less wired infrastructure requirements or even because it's impossible to have wired access due to physical restrictions in certain situations.

Wireless Local Arena Networks have been skyrocketing in popularity in the past decade. With the explosion of smartphone capabilities, everyone wants access to their information at their fingertips, ready to carry in their pockets. This has led to enormous demands resulting in immense progress in wireless technologies like 4G, 5G and WiFi. WiFi in particular has been instrumental in providing high-speed access on the cheap due to it's low cost to setup and it's use of the unlicensed spectrum, with the trade-off of being locally restricted. Initially intended for use in cashier systems, it quickly gained popularity for use with the 2.4Ghz ISM Band, which was released for unlicensed use by the FCC, leading it to be adopted very widely in households and public areas.

2.2 802.11 Protocol Family

Let us have a look at how the 802.11 protocol family has evolved over time:

The first version of 802.11 was 802.11-1997, which was released in 1997 as the name suggests. It used either Direct-sequence Spread Spectrum(DSSS) or Frequency-Hopping Spread Spectrum(FHSS) as the modulation technique. It specified a stream data rate of 1-2 Mbps over a bandwidth of 22Mhz.

The next iteration of the protocol, 802.11a introduced Orthogonal Frequency Division Multiplexing for modulation, which was a big step up and offered stream rates of upto 54 Mbps over the 5Ghz band with a bandwidth of 20Mhz.

The next major change came with 802.11n, adding capability for a 40Mhz bandwidth, and also introduced Multiple-input Multiple-output (MIMO) OFDM, which allows for beamforming on the transmitter side and diversity on the receiver side, further improving on performance.

In the next iterations, 802.11ac brought in support for wider bandwidth (80 Mhz and 160 Mhz) in the 5Ghz spectrum, and adding Multi-User MIMO (MU-MIMO). Following that, 802.11ad which is the latest version for general use, uses the 60Ghz band or millimeter waves for communication.

2.3 802.11n

IEEE 802.11n-2009, commonly known as 802.11n is the protocol we have used for the purpose of this thesis[(1).

2.3.1 OFDM PHY

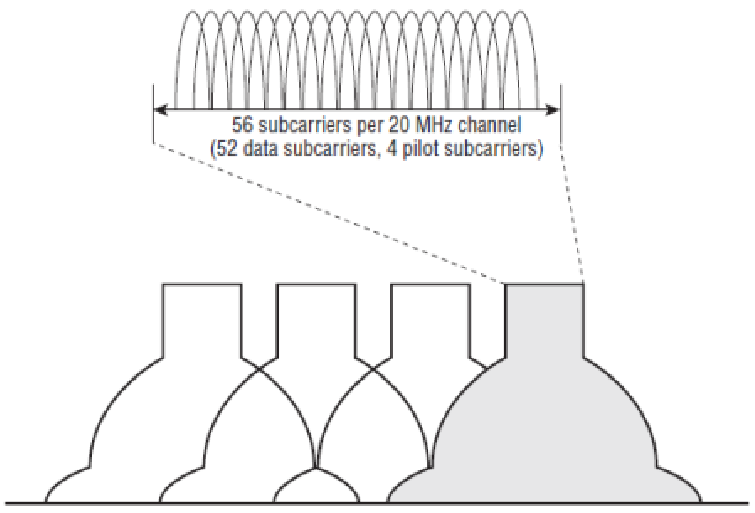
802.11n can be operated in both the 2.4Ghz ISM band as well as the 5Ghz band. In this work, we consider the 2.4Ghz band and the 20Mhz channels.

The 20Mhz channel uses an 64 point FFT, of which 56 are OFDM subcarriers. Within that, 52 subcarriers are for data, and 4 are pilot tones. for a total occupied bandwidth of 17.8Mhz. Total symbol duration is 3.6μs or 4μs, which includes the guard interval of 400ns or 800ns respectively.

In time domain, the 802.11n PHY can operate in one of 3 modes: Legacy, Mixed and Green Field mode.

- Legacy mode transmits frames in the 802.11a/g format
- Green Field mode transmits HT frames without any legacy compatible part
- Mixed mode packets have a legacy compatible part at the start, while the rest of the packet has HT frames

In terms of modulation and coding schemes, 802.11n can support upto 64-QAM modulated symbols, with a code rate of upto 5/6. Apart from this, support for upto 4 spatial streams means that the theoretical data rate limit is 288.8Mbps.



56 subcarriers per 20 MHz channel
(52 data subcarriers, 4 pilot subcarriers)

Figure 2.1: Subcarrier structure in 802.11n
Source: https://mrncciew.com/2014/10/19/cwap-802-11n-introduction/

2.3.2 MAC

MAC stands for Medium Access Control. 802.11 in general uses Carrier-sense multiple access with collision avoidance (CSMA/CA) as a multiple access method. This is used instead of collision detection because of the hidden node problem in which collision at the receiver cannot be detected by the transmitter. This method can be easily understood if we split it into two parts:

- **Carrier Sense:** Before initiating transmission, the node first senses the medium (in this case, the air medium) and checks if there is an already ongoing transmission. If the the medium is "idle", we proceed to the next ste. If the medium is busy, we wait for a random amount of time till we see an idle medium
- **Collision Avoidance:** We send a ready-to-send(RTS) message to the intended reciever. After we receive a clear-to-send(CTS) from the receiver, transmission is done. This solves the hidden node problem because CTS is sent to only one transmitter at a time.

  This process has a high overhead, so not all applications of the 802.11 protocol use RTS/CTS, depending on the use case.

After every transmission, there is a fixed interval where no one transmits called Inter-frame Spacing. This is varied based on the specific application involved.

2.3.3 Summary - improvements made in 802.11n

The design goal for the 802.11n was "HT" standing for High throughput, which it very well achieved by improving from 802.11g's maximum throughput of 54 Mbps to a possible maximum of 600Mbps on 802.11n. This was achieved by the following changes or additions:

- **More Subcarriers:** The number of subcarriers was increased from 48 to 52
- **FEC:** The highest coding rate has been increased from 3/4 to 5/6, reducing some redundancy to get better throughput from good channel conditions
- **Short Guard Interval:** A new option for decreasing the guard interval between transmission from 800ns to 400ns was added
- **MIMO:** Upto 4x4 MIMO is allowed, and provided that there are enough multipath reflections, this means a linear increase in throughput to 400%, if the receiver has multiple antennas. In the mobile phone use case, which typically have only one receive antenna, the multiple transmit antennae can be used for beamforming, providing similar results in sum throughput.

  A minimum of two antennas are mandatory for all 802.11n compliant devices, except for handsets

Other optional features like Space-Time Block Coding can improve the range for single antenna receivers, while LDPC codes can improve throughput by further improving error rates.

CHAPTER 3

Network Simulator - 3

3.1 Introduction

As described above, there are multiple nuances to wireless transmission, coming together to form a fairly complex system. This is the case with most networks, and hence to study them, it is not possible to simply run mathematical simulations based on formulas to get results which are close to real life scenarios.

Network Simulator (Version 3) is an event-driven simulator which aims to solve the problem of full system simulation for computer networks. This greatly helps networking research, since this gives an easy way to simulate the full OSI stack and get real performance results from an application level standpoint

NS-3 uses two key languages, C++ and Python, with scripting capabilities, and the simulation core supports both IP and non-IP based networks. Other capabilities like interfacing with physical devices within simulation, like computers and testbeds are also included in this powerful framework. With the version 3, support has also been added for LTE network simulation

3.2 Simulation flow

The general method to set up a simulation can be described as follows:

- **Topology definition:** Definition of the basic facilities, and their interrelationships , for example the physical air interface characteristics like path loss models, error rate models and so on. NS-3 has quite a few containers and helper classes to make this simple and easy, along with a good set of default characteristics
- **Model Development:** Most layers of the OSI protocol are managed using separate models, so that we can define exactly what we are simulating, depending on our requirements.

  Models for techniques like TCP/IP or UDP, most of the 802.11 protocol family, simple streaming client and server applications, point-to-point link models, even models for different operating systems' packet scheduling techniques are available out of the box, in the NS-3 libraries.

- **Node and link configuration:** In this part, we define the network structure with which nodes are connected and on a common network. Applications are running on these nodes. Attributes of the applications are used to set link variable values like packet size, data rate, destination address, transport protocol, and so on.

3.3 Internal simulation system of NS-3

As mentioned before, NS-3 is an event-driven simulator, which means that it does not simulate every time step and looks to see if any event happens at that point in time, but rather, has a queue of events, sorted by time and executes them one by one. Now, one event can trigger another, adding it to the queue, giving us the ability to schedule chains of events with just the first event manually programmed.

This method of event-driven simulation allows NS-3 to have a very high time resolution of upot 1ns while not sacrificing computing power. Apart from this, the models defined for each component of the network protocol allow it to represent transmission to a fair degree of accuracy without going into the process of actually modulating randomly generated packets, then doing processes like channel estimation, error correction and so on. This whole process is substituted by models for error rate (for example Nist Error rate model, which is the default used in our simulation), which depend on the parameters of transmission like distance and Tx power and MCS selection (which is also dynamically selected by the use of Rate Managers). Therefore it gives a good balance between simulation speed and realism.

CHAPTER 4

Simulation parameters

The scenario we have planned is one in which an operator wants to provide WiFi access to users in a local area, with a dense deployment of APs and Users.

4.1 Node Definition

The basic unit of our simulation is a node. A node can be any device on a network, an AP or an STA. Since we will be have a very large number of nodes in our simulation, we need a way to handle them in batch. Fortunately, NS-3 has an easy way to achieve that with the helper class `NodeContainer`. We define our nodes in two separate categories for ease of understanding - AP nodes and STA nodes, like as follows:

```
NodeContainer wifiApNodes;
wifiApNodes.Create (nAp); // nAp = number of APs
NodeContainer wifiStaNodes;
wifiStaNodes.Create (nStaPerAp*nAp); // nStaPerAp = number of
   users per AP
```

This creates two node container objects - one for the APs and one for the user devices. Now we can access any node through the container object using: `wifiApNodes.Get(n)`, similar to how one would access objects in an array.

4.2 PHY and channel definition

Next in the simulation, we shall define the physical layer characteristics, which is done by the following:

```
YansWifiChannelHelper channel = YansWifiChannelHelper::Default
   ();
```

```
YansWifiPhyHelper phy = YansWifiPhyHelper::Default ();
phy.SetPcapDataLinkType (WifiPhyHelper::DLT_IEEE802_11_RADIO);
phy.SetChannel (channel.Create ());
```

YANS stands for "Yet another network simulator" and is one of the most widely used helper classes for NS-3. The benefit of using a channel helper is that most of the configuration for getting a WiFi PHY layer configured would be a lot of boilerplate code, which is unnecessary for most use cases. Plus, the helper classes are developed with lots of consideration and research going into the defaults, so that the approximations made in order to simulate the network are closest to the real world.

MIMO and Spatial Multiplexing

Next, to make full use of the capabilities that 802.11n has to offer, we have to make a few changes to the default configurations. For example, to enable MIMO or spatial multiplexing we must have:

```
  // Set MIMO capabilities
phy.Set ("Antennas", UintegerValue (nStreams));
phy.Set ("MaxSupportedTxSpatialStreams", UintegerValue
   (nStreams));
phy.Set ("MaxSupportedRxSpatialStreams", UintegerValue
   (nStreams));
```

Since 802.11n allows upto 4 streams, we set the `nStream` variable to 4. By default, it is always set to 1.

Transmit Power Control

The PHY definition is also where we must define the maximum and minimum transmit powers for the devices and APs on the network. To do that, we do the follows:

```
  // Set Transmit Power Control
phy.Set("TxPowerStart", DoubleValue(txPower));
phy.Set("TxPowerEnd", DoubleValue(txPower));
```

Transmit power is a very sensitive variable, since it will decide how much interference we will encounter on our receivers. In our experiment, we shall vary the transmit power as a variable and see how throughput varies. Also, Tx Power can be more finely controlled, with different policies on APs and STAs, depending on the present channel conditions, and the distance to the receiver, but we shall ignore that and have one constant setting for all.

Other defaults for PHY and Channel:

- The Error rate model is also defined as a component of PHY, and by default it is the NIST Error rate model[(4)].
- The Channel object has two main properties which are important for transmission:

  Propagation Delay model - this varies based the medium in which communication is being conducted. The default value is for air and is the delay is inversely proportional to the speed of light.

  Propagation Loss model - This is also one of the more important factors to consider in simulations, and the default model is the Log Distance Loss model, which is fairly accurate

Wifi Protocol and Rate Manager Setup

Next we set up the WiFi helper, which helps in binding the PHY and MAC protocols to the AP and STA devices. First we instantiate the WiFi helper object as follows:

```
WifiHelper wifi;
wifi.SetStandard(WIFI_PHY_STANDARD_80211n_2_4GHZ);
wifi.SetRemoteStationManager ("ns3::MinstrelHtWifiManager");
```

Minstrel-HT Wifi Manager seen above is a rate adaptation mechanism for the 802.11n/ac standards. It dynamically learns the rates that can be supported on the channel and manages things like RTS/CTS threshold, number of retransmissions to try for a particular packet, and many others.

4.3 Physical layout

Consider that the wired backhaul lines are along a road, so an we are assuming the deployment would be along that road, on both sides of a road, assuming a width of 30m for the road. APs will be placed regularly, 15m apart.It is assumed that the STAs which are randomly dropped, will be connected to their nearest AP. The scenario will look as below:

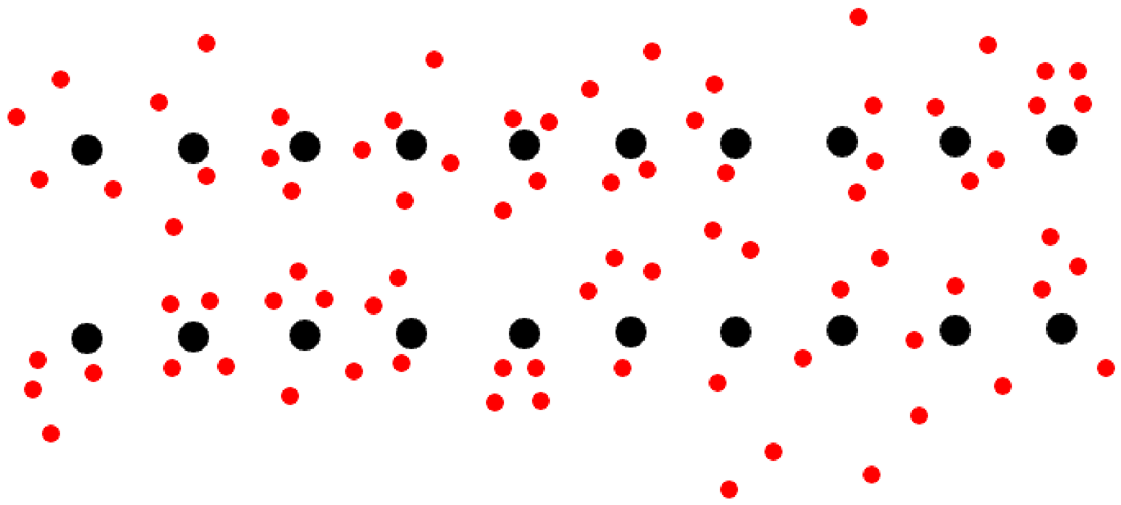


Figure 4.1: Positions are allocated in this fashion. The APs (black), are regularly located, while STAs (red) are randomly placed

These physical location to be simulated can be defined by the use of Mobility Models.

Mobility Models

There are various mobility models available in NS-3, from stationary position to linear movement to random path models, to simulate pedestrian motion, etc. Below is how we setup our mobility model in the simulation.

```
MobilityHelper mobility;
Ptr<ListPositionAllocator> positionAlloc =
   CreateObject<ListPositionAllocator> ();
mobility.SetMobilityModel
   ("ns3::ConstantPositionMobilityModel");
```

This creates a static positioning model. `positionAlloc` is a helper object which will make it easy to assign positions for our devices.

```
mobility.SetPositionAllocator (positionAlloc);
```

```
positionAlloc->Add (Vector (x, y, z));
mobility.Install (wifiNodes);
```

Each desired position given in (x, y, z) coordinates is added in sequence into the position allocator object and then this allocator will assign positions to the nodes contained in the `wifiNodes` container in order.

4.4 SSID and IP address allocation

Each AP has a unique SSID for it's network and has it's subnet in the 192.168.X.0 format. Again, NS-3 makes it simple by providing IP Address Helpers and Interface containers for ease of use. We need separate containers for interfaces rather than simply using the device containers because one device might be configured to have multiple interfaces, to communicate with different subnets.

Address assignment works by starting with a base address for the subnet. Following this, each interface is added to the subnet one by one and is assigned an IP address in sequence.

Routing

Routing is provided by an Internet Stack. To define the routing protocols and to aggregate IP/TCP/UDP functionality to nodes, we can use the InternetStackHelper, and this is done in our simulation. It is to be noted that this step is not significant in any way, since our focus is on link throughput maximization, and there is no data transfer which is longer than one hop.

The Internet Stack also can be used for pcap tracing (Packet Capture Tracing), if required.

4.5 Applications

The application layer is where we shall have the meat of our simulation, defining the Transport Protocol (TCP or UDP) to be used for simulation, as well as parameters like packet sizes, packet arrival behaviour, data rate for input streams for both Uplink and Downlink transmissions. A simple application for one directional communication on the downlink (packets going from AP to STA) will be as follows:

```
uint16_t port = 5001;
UdpServerHelper server;
server.SetAttribute("Port", UintegerValue(port));

ApplicationContainer serverApp;
serverApp = server.Install (wifiStaNode);
serverApp.Start (Seconds (0.0));
serverApp.Stop (Seconds (simulationTime + 1));
```

In this block we first set up the UdpServerHelper object. We set the port to be used as port 5001. In the next part, we create the server application by calling "install" on the helper and passing the STA node as an argument. The advantage of having the helper configured separately from the node is that now this same helper object can be used to install on various nodes, without having to repeat yourself.

```
ApplicationContainer clientApp;
InetSocketAddress dest(port);
dest.SetIpv4(StaInterface);

OnOffHelper client ("ns3::UdpSocketFactory", Address());
client.SetAttribute ("Remote", AddressValue(dest));
client.SetAttribute ("OnTime", StringValue
   ("ns3::ExponentialRandomVariable"));
client.SetAttribute ("OffTime", StringValue
   ("ns3::ExponentialRandomVariable"));
client.SetAttribute ("DataRate", StringValue ("100000kb/s"));
client.SetAttribute ("PacketSize", UintegerValue
   (payloadSize));
```

```
clientApp = client.Install (wifiApNode);
clientApp.Start (Seconds (1.0));
clientApp.Stop (Seconds (simulationTime + 1));
```

In the second part we configure the client application, which will send packets to the IP address and port, which is given as the destination using the InetSocketAdress object.

After that, we define the packet arrival behavior within the client application, using the On-Off traffic generation model. In this model, the client will stay 'on' and output packets for a time given as `onTime`, and stay 'off' for a time given as `offTime`. These are typically given by random variables to simulate real-life transmission patterns, and exponential and pareto distributions are commonly used. While the client is on, it outputs a constant data rate, with each packet being `payloadSize` in bytes.

4.6 Data Processing and Packet Capture

NS-3 provides a very user-friendly API for extracting meaningful information, gathering insights and for debugging purposes.

One of the most widely used techniques is to generate trace files of all the packet transmissions and store them in a dump file. This can be processed and analyzed later using utilities like tcpdump or Wireshark.

Since we are only interested in the throughput aspect of the of the simulation, we use packet tracing only for debugging purposes. Throughput details can directly be gotten from the Server Application by the following:

```
totalPacketsThrough = DynamicCast<UdpServer> (serverApp.Get
    (0))->GetReceived (); \\UDP
totalBytesThrough = DynamicCast<PacketSink> (serverApp.Get
    (0))->GetTotalRx (); \\TCP
```

This allows us to directly extract the values we need from the simulation environment right after NS-3 finishes simulation.

CHAPTER 5

Simulation Results

We run our simulations in a few different scenarios, with the following parameters common across all scenarios:

- Number of APs: 20
- Number of STAs per AP: 4
- Channel: Center Frequency = 2432Mhz, Bandwidth = 20Mhz
- Packet(Payload) Size: 1472 bytes
- Simulation Time: 10 sec
- Maximum number of MIMO/spatial streams: 4
- RTS/CTS: enabled
- Packet arrival behavior: Exponential On-Off

Apart from this, we have 2 parameters where we have choices leading to different instances:

- Transport Protocol:
  TCP
  UDP
- Traffic mixture
  100% Downlink traffic
  70% Downlink + 30% Uplink traffic

With the above variables, we also include Transmit power control and observe how a change in transmit power changes the sum throughput of the network.

For reference, we have also calculated the maximum throughput for a 1 AP, 1 STA scenario. In this case, power control is irrelevant in terms of interference, so we transmit at a high power of 20dBm to maximise throughput.

	Downlink	Downlink + Uplink
TCP	143.776 Mbit/s	147.994
UDP	208.321 Mbit/s	205.541 Mbit/s

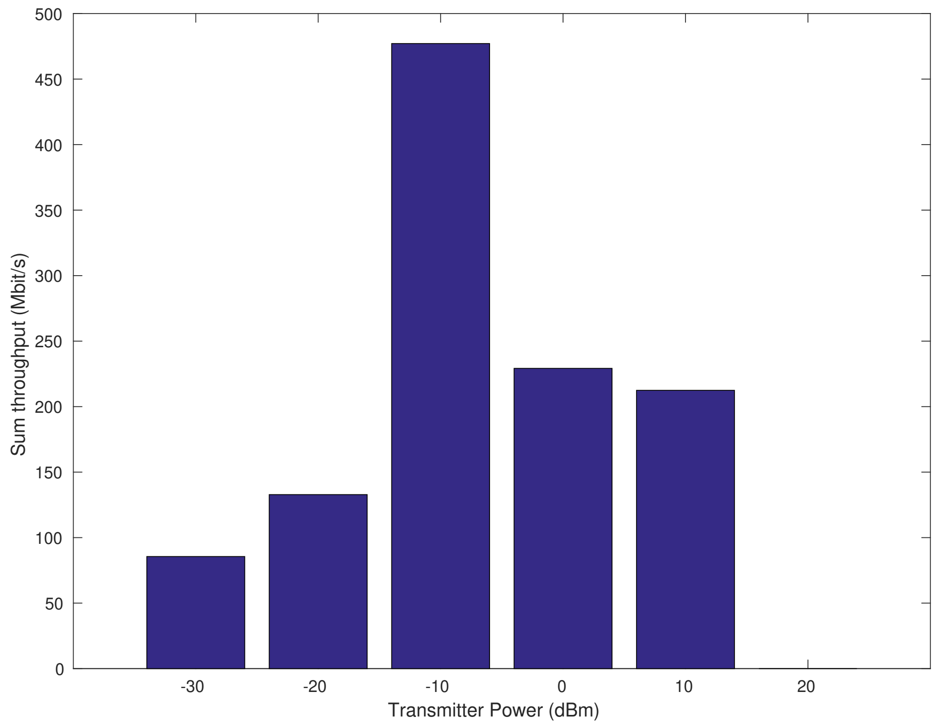


Figure 5.1: UDP - Downlink Only

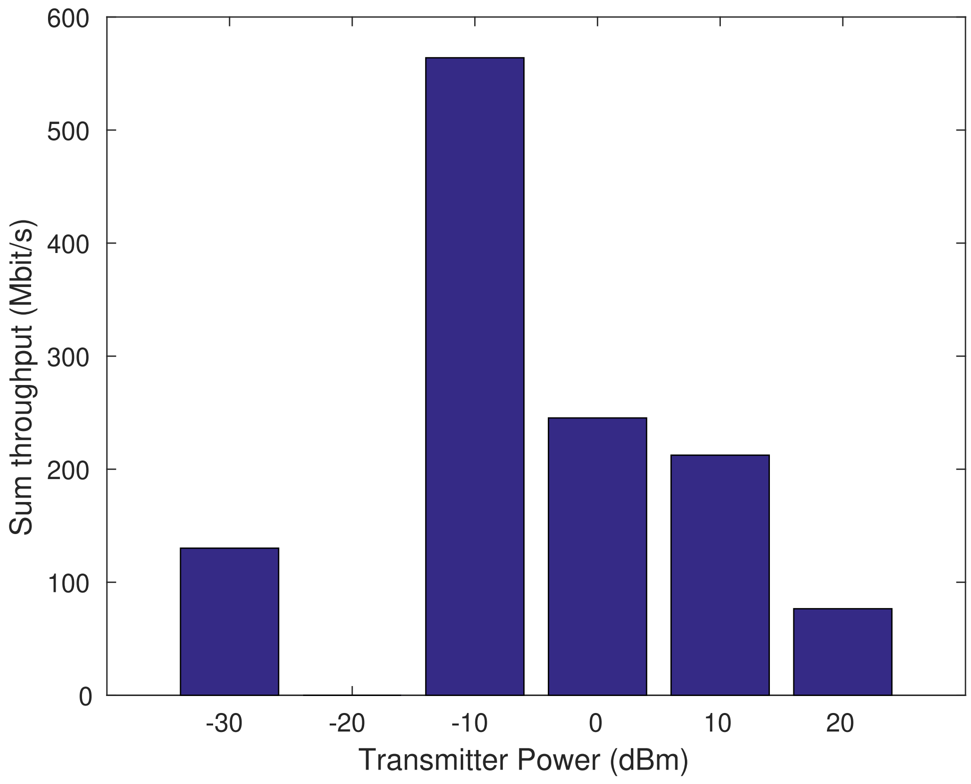


Figure 5.2: UDP - Uplink and Downlink

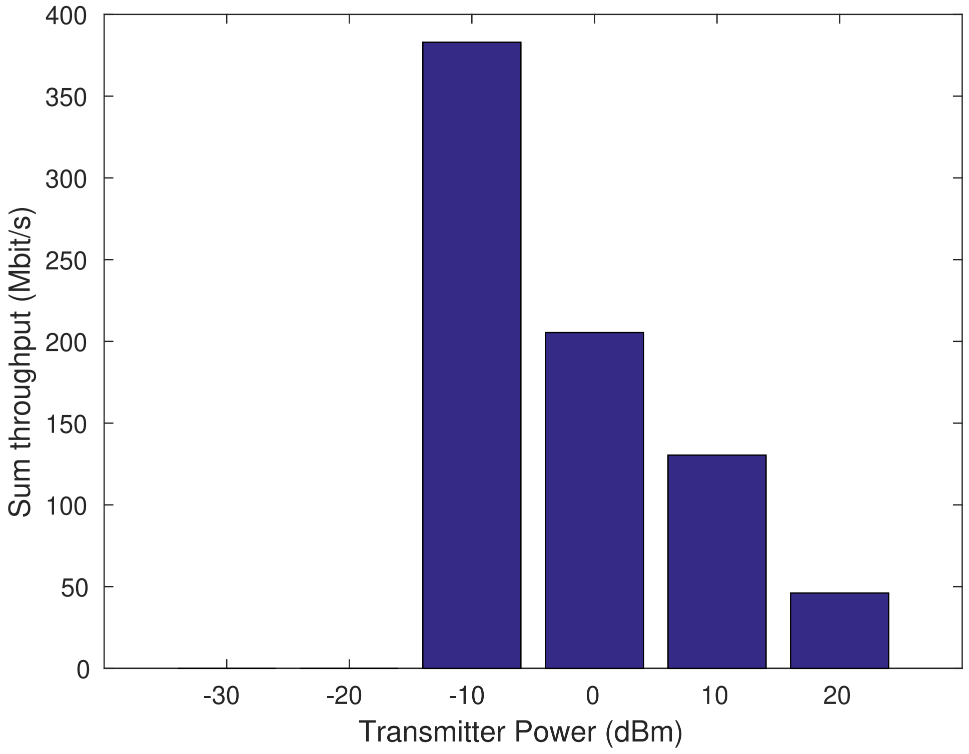


Figure 5.3: TCP - Downlink

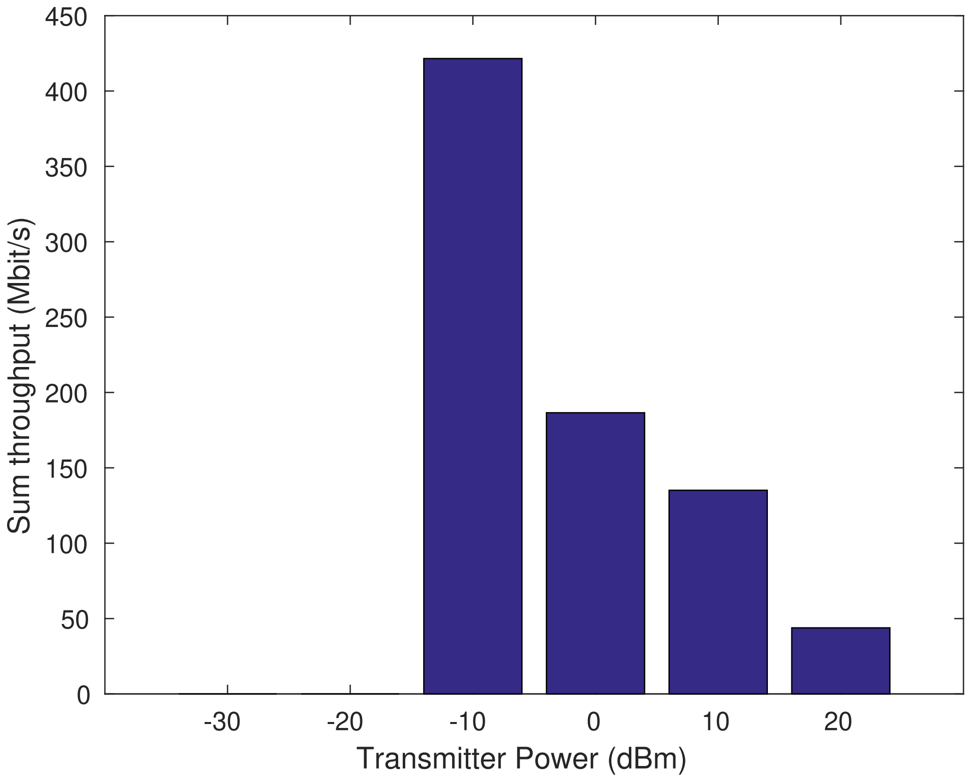


Figure 5.4: TCP - Uplink and Downlink

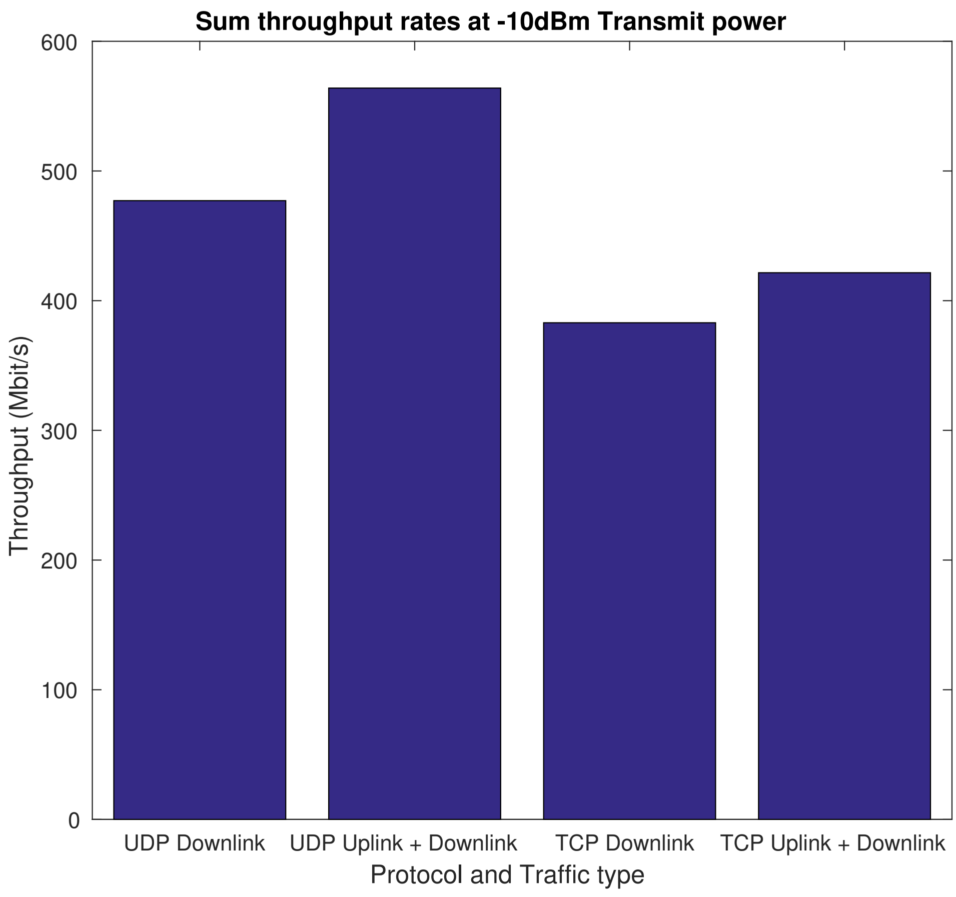


Figure 5.5: Best Transmit power scenario comparisons

CHAPTER 6

Conclusions and Future Work

The conclusions we derive from the results shown are:

- The columns where throughput is shown as 0 are the cases in which some links fail to successfully establish. This is generally due to very low SINR, which causes transmissions to fail repeatedly
- The performance difference between TCP and UDP is as expected, with TCP slightly lower than UDP because of the protocol requirement of ACKs for every successful transmissoin
- When applied carefully, transmit power control can easily help us extract 2x to 4x more than the single link throughput, on the same channel
- The figures comparing Downlink only and Uplink + Downlink are interesting, showing us that because we use CSMA/CA, both the APs and STAs contend equally to transmit on the channel, and having the Uplink component actually gives us an improvement that is not insignificant.
- This result is for a case where the STAs are closer to the AP than other devices not on the network. We think that as the required range increases due to farther APs, the transmit power requirement will cause higher interference between adjacent networks, giving us results similar to what we see in the 10dBm or 20dBm transmit power cases

  This means that to get more than 100% channel capacity on the network, we may need to introduce multiple channels and have a deployment similar to the Frequency Reuse deployments we see in cellular systems

Future work along this line of application can be done in:

- Considering a way of arranging co-operation among APs to minimize interference locally, allowing for better data rates.
- Moving the application of offloading to local networks into a licensed band, allowing us to replace CSMA/CA, which is quite inefficient with a better MAC protocol
- Using this system in a low frequency spectrum (like 900Mhz or 450Mhz) will yield better coverage. The caveat is that NS-3 does not have native support for 450Mhz, and 900Mhz is only supported in 802.11af, so to use NS-3, we must go into editing the inbuilt libraries.

CHAPTER 7

Other Work - Rayleigh Fading Channel Generator Using the Dent Model

Channel estimation is a crucial part of communication system design. One characteristic of non-Line-of-Sight(nLOS) channels with large number of scatterers is rayleigh fading behaviour. This is very common in urban scenarios, with many tall buildings with flat surfaces to reflect signals well.

It's mathematical formulation is given by

$$R = X + iY$$

where X and Y are uncorrelated random variables. How rapidly the channel shifts is a function of the doppler shift f_d or the relative velocity between the transmitter and receiver. Due to this time-shifting nature, the rayleigh fading waveform is time-correlated.

Jakes' model [(5)] is a deterministic method to generate time-correlated Rayleigh fading waveforms. It uses a sum of independent oscillators to generate the waveform.

However, when generating multiple such fading waveforms for a multi-tap channel, we see that there is a significant correlation between the different channel taps.

In the Dent model [(6)], the proposal is to slightly alter the angles of arrival assumed by Jakes' model, and more importantly weight the different oscillators used to get the final channel gain, with orthogonal functions, which in this case are Walsh-Hadamard codewords. This gives a significant decrease in cross-correlation (in the order of 10^5).

The above described model was implemented in MATLAB, to be used as the channel generator for all subsequent research work pursued by our lab.

REFERENCES

[1] *IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput*. IEEE Std 802.11n-2009 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, and IEEE Std 802.11w-2009).

[2] Bose, J. C. (1899). *On a Self-Recovering Coherer and The Study of the Cohering Action of Different Metals*. Proceedings of the Royal Society, LXV(no. 416), 166-172. ISSN S 0018-9219(98)00763-4.

[3] Molisch, A. F., *Wireless communications, volume 34*. John Wiley & Sons, 2012.

[4] Pei, G. and Henderson, T. R. (2010). *Validation of OFDM error rate model in ns-3*. URL `https://www.nsnam.org/~pei/80211ofdm.pdf`

[5] Jakes, W. C., *Microwave Mobile Communications*. Wiley-IEEE Press; 2nd edition (May 16, 1994).

[6] Dent, P. and Bottomley, G. E. and Croft, T. (1993). *Jakes fading model revisited*. Electronics Letters, vol.29, no.13 pp. 1162-1163

CURRICULUM VITAE

1. NAME	: Deepak Padamata
2. DATE OF BIRTH	: 19 October 1995
3. E-MAIL	: deepakpadamata@gmail.com
4. Work Experience	: Amazon Development Center India (2016) Software Development Engineer Intern