Analysis of Transport Layer Congestion Control Algorithms over 5G Millimeter Wave Networks

F. Siddiqui
Dept. of Mathematics and Computer Science
Dickinson College
Carlisle, PA
siddiquf@dickinson.edu

Q. Chau
Dept. of Mathematics and Computer Science
Dickinson College
Carlisle, PA

Abstract**—** The Millimeter Wave technology can provide very high data rates and is a key enabler of 5G communication. However, mmWave signals suffer with high penetration loss and poor isotropic propagation which causes intermittent packet losses. TCP's congestion control algorithms consider packet loss as an implicit notification of network congestion and react by reducing the data transmission rate. In this research we examine how TCP's congestion control algorithms impact the achievable data rate over mmWave links. We discuss the performance of different TCP versions using metrics such as congestion window size (cwnd), throughput, Round Trip Time (RTT), and Signal-to-Interference-plus-Noise Ratio (SINR).

Keywords— TCP, mmWave, Congestion Control, ns-3

I. Introduction

Millimeter Wave (mmWave) [5][7] is a communication technology that uses the underutilized radio frequency spectrum (Extremely High Frequency - EHF), which ranges between 30 GHz and 300 GHz with a wavelength of 1-10 millimeter. mmWave technology supports wider bandwidths (higher data transmission rates) than most current wireless commercial systems that use the low frequency spectrum (frequencies below 6 GHz). To fully utilize the mmWave technology in mobile access networks, further developmental efforts are needed to overcome issues such as high penetration loss and poor isotropic propagation as described below.

1. High penetration loss: Millimeter waves are vulnerable to physical things such as building walls, human beings, etc. and suffer from considerable attenuation passing through foliage. mmWave signal propagation is also affected by rain, water vapor and atmospheric gases [1].

2. Poor isotropic propagation: Isotropic propagation allows signals to travel in all directions. Poor isotropic propagation limits the distance between the sender and the receiver. Although mmWave technology provides high-frequency bands and allow devices to transfer more data in less time, it has a small coverage area (about 100 meters for a single mmWave base station [2] compared to current cellular technologies (such as 4G-LTE which has a coverage radius of a few kms). Thus, more mmWave base stations are needed to provide coverage in a large area.

Poor isotropic propagation and high penetration loss are the main issues in mmWave networks. They can possibly cause weak, inconsistent network connections, delays, and packet losses. TCP's congestion control mechanism interprets these intermittent packet losses as network congestion and limits the amount of data the device can transmit. This can result in low transmission rates and therefore poor utilization of mmWave's high bandwidth. The TCP protocol has different operating versions, each of which utilizes a slightly different congestion control algorithm.

In this work we will evaluate the performance of different TCP Congestion Control algorithms over mmWave communication links. The algorithms are simulated by using five different TCP versions (New Reno, YeAH, Hybla, Westwood, and Vegas). The Congestion Control algorithms supported by these TCP versions differ in their reactions to missing acknowledgement packets and to duplicate acknowledgement packets. The algorithms also vary in their calculations of the congestion window *(cwnd)* size after experiencing packet loss and delay. It is worth noting that among the TCP versions that we selected, New Reno, Westwood and Hybla are loss-based algorithms, while Vegas is a delay-based algorithm and YeAH is a hybrid algorithm. Loss-based algorithms detect network congestion by observing packet loss. On the other hand, delay-based algorithms use delay as a congestion indicator, meaning that an increase in Round-Trip-Time (RTT) will cause them to adjust the congestion window *(cwnd)* size. Hybrid algorithms use both loss and delay as congestion indicators. Their goal is to detect network congestion before the network queues are completely full and packets need to be dropped. We present the simulation results of the performance of Congestion Control (CC) algorithms in different scenarios. The efficiency of the CC algorithms when used in mmWave networks is assessed by using various network performance metrics such as congestion window *(cwnd)* size, throughput, Round-Trip-Time (RTT), and Signal-to-Interference-plus-Noise-Ratio (SINR).

The rest of this paper is organized as follows. Section 2 describes the various entities set up in the simulation environment to conduct this study. Section 3 discusses the performance metrics used to assess the performance of the Congestion Control algorithms. In section 4 we describe the simulations of various TCP Congestion Control algorithms and discuss the observed performance. Finally in section 5 we make some concluding remarks.

II. Simulation Components

To investigate the performance of TCP in mmWave networks, we use the Network Simulator 3 (NS-3). NS-3 is a discrete-event network simulator for Internet systems [10], which provides predefined models and data traces needed to set up an end-to-end network simulation. The NS-3-mmWave module is built on top of NS-3 to support end-to-end simulations for mmWave network links. The end-to-end simulation of a mmWave network will include the following network entities:

- **Evolved Node B (eNB):** eNB or the Base Station (BS) connects the mobile access network to the backhaul core network. A wireless (mmWave) communication link exists between the mobile device and the eNB.
- **User equipment (UE):** UE is the user device with a mmWave radio interface that uses the mmWave mobile access network to communicate with any other mmWave device or a with a server on the TCP/IP wired network (public Internet).
- **The Internet:** The Internet is simulated by creating a wired network with a specific data rate, Maximum Transmission Unit (MTU) and the propagation delay of each link created. The Internet will include devices (gateway, hosts, etc.) with IP addresses assigned to them.
- **Network Core:** The network core consists mainly of two components: The Packet Data Network Gateway (PGW) and the Mobile Management Entity (MME). The PDN Gateway provides access to external Packet Data Networks for the UE, by being the point of exit and entry of traffic for the UE. PGW acts as an IP router with support for specific tunneling and signaling protocols. A MME keeps records of the eNBs (or base stations) in a specific area [9].
- **Remote host:** Remote host is a remote server with one or more network applications running on it. A TCP/IP stack is installed on the remote host to communicate with the UE through the PDN Gateway and the eNBs.
- **Obstacles:** Obstacles can be human beings, buildings, bridges, or other physical objects. Most obstacles are given a size and a constant position such that they obstruct the mmWave propagation.
- **Channel Conditions**: There are two channel conditions: Line of sight (LOS) and Non-line of sight (NLOS).
  - **Line of Sight (LOS):** When the channel condition is LOS, there are no obstacles residing between the transmitting and receiving antennas, meaning that waves travel in a clear path from the user device (called as the UE) to the Base Station.
  - **Non-Line of Sight (NLOS):** When the channel condition is NLOS, there is full or partial obstruction existing between the transmitting antenna and the receiving antenna. Obstacles that commonly cause NLOS condition are buildings, trees, human body, and possibly rain.

Typically, an NLOS condition implies that there should be an alternative path chosen for transmission. However, it is worth noting that NLOS condition does not necessarily indicate the complete loss of network connection. Instead, NLOS lowers the effective signal strength, which reduces the chance of a successful transmission. In addition, it is also possible to configure the physical (PHY) and Medium Access Control (MAC) layers of UEs and eNBs.

III. Performance Metrics

The main performance metrics that will be used to assess the TCP performance over mmWave networks include the following:

- **Congestion Window *(cwnd)* Size (in bytes):** The congestion window is a variable set by Congestion Control algorithms to limit the amount of unacknowledged data being sent into the network. Each CC algorithm modifies the congestion window differently when packet losses and delays are experienced. Investigating the *cwnd* sizes provides an idea about the utilization of network bandwidth by the user.
- **Throughput (in Mbps):** Throughput is the rate of successful packet delivery over a communication channel. We compute throughput by calculating how many Megabits have been delivered to the receiver from a sender per second. Although the bandwidth for a network link remains unchanged, throughput can be affected by many factors such as delay (network latency), network congestion and high network traffic.
- **Round Trip Time (RTT, in seconds):** the amount of time it takes for a packet to be sent from the sender to the receiver plus the amount of time it takes for the acknowledgement of the packet to be sent from the receiver to the sender. Most congestion control algorithms update the congestion window size every RTT.
- **Signal-to-Interference-plus-Noise Ratio (SINR, in dB):** SINR is a metric used to compute the strength of a wireless signal. SINR is calculated using the equation $\boxed{SINR = \frac{P}{I+N}}$ where P is the power of incoming signal of interest, I is the interference power of interfering signals in the network, and N is some noise constant. The value of SINR is typically positive when the channel condition is in LOS. When the channel enters NLOS, the value of SINR becomes negative. The network is detected to be in outage if SINR goes below the outage threshold, which is -5 dB in our simulations.

IV. Simulations and Discussion of Results

A. Simulation 1a

Our first simulation is a use case of the *UMi-StreetCanyon* scenario. *UMi-StreetCanyon* represents dense urban areas where the Base Stations (BSs) (also known as eNode B (eNB)) are placed below the rooftop of the surrounding buildings.

Table 1: Simulation 1a configuration

Network Component	Parameter	Value
3GPP Channel	Channel Scenario	Urban Microcell

TCP	TCP Maximum Segment Size (MSS)	1000 bytes
	Sender/Receiver Buffer Size	5 MB
UE	Number of UEs	1
	Speed	1.4 m/s (walking speed)
	# of packets	5000000
Application (installed at UE)	Packet size	900 bytes
	Data rates	1 Gb/s
Other	RLC Buffer size	1 MB
	Carrier Frequency	28 GHz
	Direct Beamforming	True
	mmWave SNR Outage Threshold	-5 dB

In this first simulation, we aim to simulate a scenario where the channel condition moves from Line-Of-Sight (LOS) to Non- Line-Of-Sight (NLOS) and back to LOS condition. This simulation is set up such that the UE moves (using the *ConstantVelocityMobility model* in NS-3 [6]) from a position where the mmWave channel condition is LOS to a position where the channel condition is NLOS. Specifically, the wireless signal between the mobile node (called the User Equipment (UE)) and the mmWave eNB/BS are not blocked initially, but they get blocked by a building by the time we cannot draw a straight line between the UE and the mmWave eNB (figure 1). Additionally, before the UE moves, it stays at the initial position for one second to demonstrate the stable throughput when the channel condition is LOS. The values of the configuration parameters used in this simulation are shown in table 1.

Figure 1 demonstrates the positions of each component in the scenario in a 2-D graph. The value *h* indicates the height of the component in meters. Following the requirement of the *UMi* scenario, we set the height of the mmWave eNB lower than the height of the building obstacle. The shaded area represents the area where the channel condition enters NLOS, meaning that we cannot draw a straight line between the UE and the mmWave eNB without going through the building obstacle.

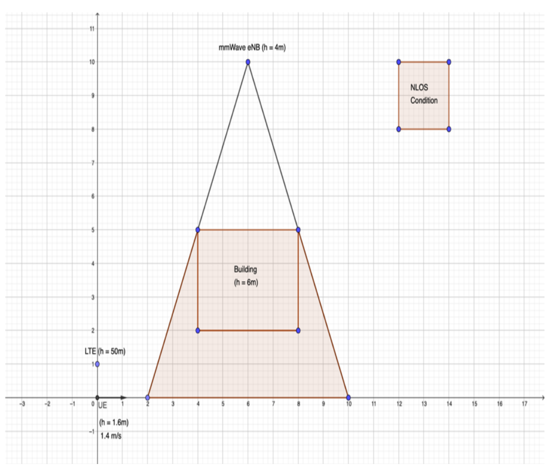


Figure 1: Positions of Network Components in simulation 1a

We now discuss the behavior of the Congestion Control algorithms when the UE moves slowly (at walking speed) and transitions between LOS and NLOS channel conditions due to the presence of a building obstacle. Initially, the channel condition is LOS as there is no blockage between the UE and the mmWave eNB. At 2.5 seconds, the channel condition enters NLOS, which causes a drop in the SINR values during the next few seconds (from around 2.5 to 7.5 seconds as shown in figure 2).

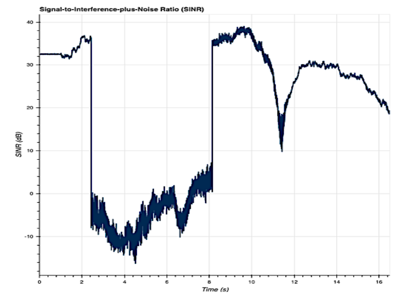
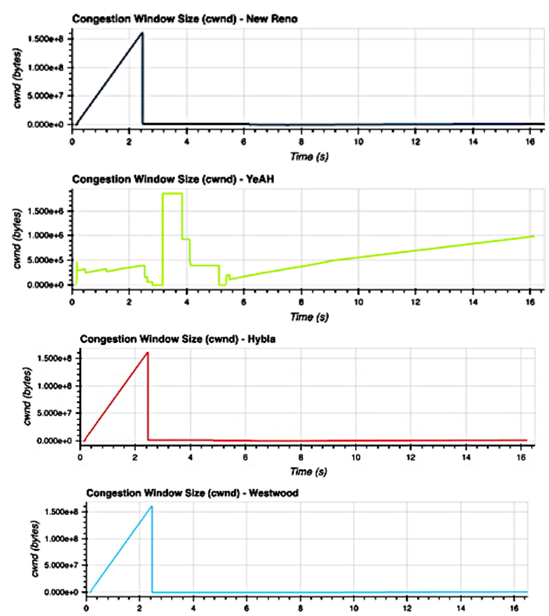


Figure 2: Signal-to-Interference-plus-Noise Ratio recorded in Simulation 1a

We notice that during the first LOS period (at approximately seconds 0 – 2.5), the congestion window of some TCP versions (New Reno, Hybla, and Westwood) goes up to approximately 1.5×10^8 bytes. This is because the initial slow start threshold (ssthresh) used in these algorithms is 2^32-1. In the Congestion Control algorithm's slow start phase, if the UE does not experience any congestion event (such as packet loss or duplicate acknowledgements, or a high RTT), the congestion window increases exponentially until it reaches the ssthresh. ssthresh is a threshold such that if the congestion window exceeds this threshold, the algorithms enter the Congestion Avoidance phase, in which the congestion window is additively increased by one MSS every RTT. This behavior is illustrated in figure 3.



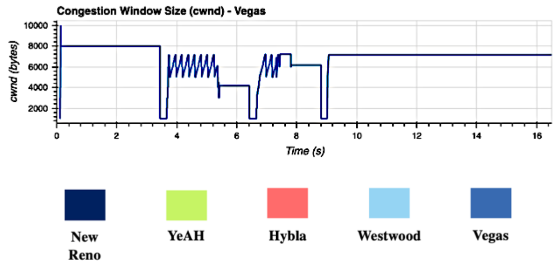


Figure 3: Congestion Window Size *(cwnd)* values observed in Simulation 1a

We computed the *uplink throughput* (figure 4) for the UE based on the number of packets received at the remote host that are sent from the UE. To do this, we install a *sink application* at the remote host. The role of a sink application is to consume the traffic generated to the IP address and port assigned to this application. The application running on the UE is configured to send packets to this IP address and port number.

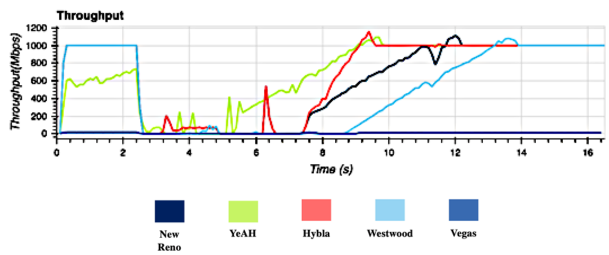


Figure 4: Uplink throughput generated by Simulation 1a

Among the TCP versions we chose for this project, Vegas is the only delay-based algorithm. Delay-based algorithms observe variations in RTT to infer the backup in router queues [3]. While delay-based algorithms detect congestion through any increments in RTT, a steady RTT indicates that the network is in a congestion-free state. TCP Vegas does not have slow start and congestion avoidance phases like loss-based algorithms. Instead, it adjusts *cwnd* by calculating the number of packets queued ($\boxed{\Delta}$) for every RTT based on the actual and expected transmission rate. The expected transmission rate is computed by $\boxed{\frac{cwnd}{RTT_{min}}}$, and the actual transmission rate is computed by $\boxed{\frac{cwnd}{RTT_{current}}}$. The algorithm tries to keep the number of packets in queue within a predefined threshold (*alpha* and *beta*). If $\boxed{\Delta > beta}$, then the congestion window is decreased by 1 MSS. If $\boxed{\Delta < alpha}$, then the congestion window is increased by 1 MSS. Since in each RTT, the *cwnd* is only adjusted by 1 MSS, the growth of the congestion window in TCP Vegas is significantly low, leading to the algorithm not fully utilizing all the available bandwidth. We can see this in figure 4, where the throughput produced by TCP Vegas is much lower than the throughput produced by other TCP versions (around 0 - 20 Mbps). In *NS-3*, the default values of *alpha* and *beta* are 2 and 4 respectively (the same values are used in Linux implementations), which means the algorithm will try to keep the buffer queue size between 2 packets (2000 bytes) and 4 packets (4000 bytes). As stated in table 1, our buffer size is 5 MB ($\boxed{\sim 5{,}000{,}000}$ bytes). Therefore, the network resources are not fully utilized, causing low throughput and small congestion window size. In addition, TCP Vegas is known for achieving high throughput in wired networks [4] but the fixed *cwnd* increase rate does not seem to work very well in mmWave networks. When mmWave is used as the radio link, delays, or long RTTs, may happen regularly due to poor signal power rather than network congestion at routers. Because of poor signal power, retransmissions at a lower layer (e.g. MAC layer) are needed without the knowledge of the transport layer. As a result, TCP Vegas interprets the increase in RTT due to lower-layer retransmissions as delays and keeps the congestion window low.

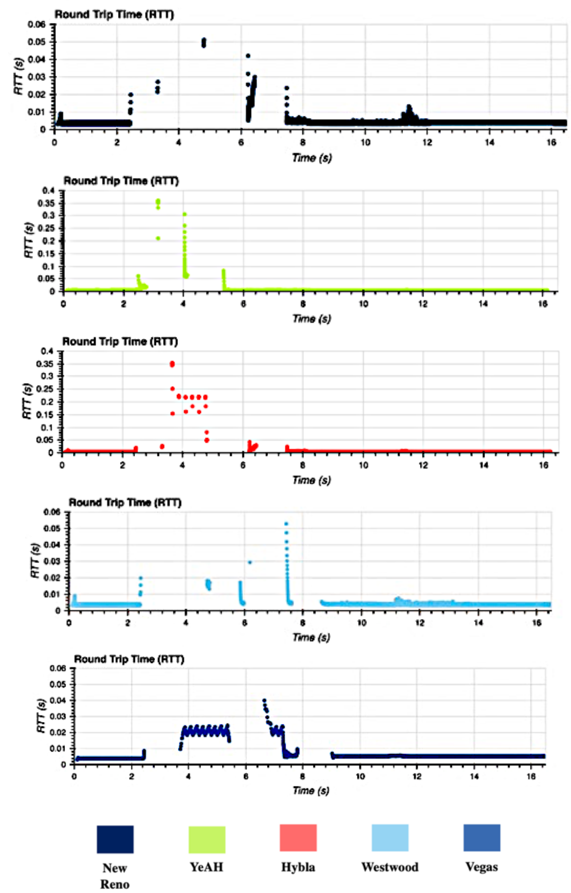


Figure 5: Round Trip Time (RTT) generated by Simulation 1a

The hybrid algorithm that we analyze in this study is TCP YeAH. Hybrid algorithms detect network congestion based on both loss and delay (RTT). In figure 5, at around 0.2 seconds, we can notice in New Reno, Westwood and Hybla (all loss-based algorithms that detect congestion solely based on packet losses) a slight surge in RTT is observed, but we cannot see this in TCP Vegas and TCP YeAH. This is because Vegas and YeAH consider an increase in RTT (delay) an indicator of congestion, and it modifies the congestion window to keep the RTT steady even if there is no packet loss. Therefore, for YeAH and Vegas in figure 3, there is a drop in *cwnd* at that exact point of time, while the value of *cwnd* keeps increasing in the graphs of the other three TCP versions. For YeAH, because of this drop, it exits the *slow start* phase, meaning that the *cwnd* no longer increases exponentially. As a result, the throughput produced by TCP YeAH could not reach the maximum value (1,000 Mbps) during the first LOS period.

TCP New Reno, TCP Westwood and TCP Hybla have *loss-based* congestion control algorithms, which means packet loss is the only indicator of congestion. Overall, during the first LOS period, we can see that the loss-based algorithms achieve higher throughput than the other algorithms since no packet losses are experienced in the initial LOS stage. Since there is only one obstacle in this simulation, the congestion window size is not expected to drop frequently using any loss-based algorithm.

During NLOS, in figure 5, among the loss-based algorithms, the RTT values of Hybla are high and recorded more frequently than the New Reno and Westwood. This shows that even in NLOS, TCP Hybla was still able to handle the transmission of some packets. In addition, after the NLOS period, TCP Hybla ramped up the congestion window the fastest among all algorithms and achieved a higher throughput. Because our simulations are set up for mmWave networks, in which long RTTs can be experienced during NLOS periods, Hybla shows better performance than Westwood or New Reno in terms of congestion window growth rate after the NLOS period ends and the UE enters the LOS channel state. In fact, TCP Hybla keeps a value called RTT_0 (the minimum RTT observed during a connection). Then the algorithm computes a value $\boxed{\rho = \frac{RTT}{RTT_0}}$. During the *Fast Recovery* phase, the equation calculating the congestion window growth considers the value of $\boxed{\rho}$ to compensate for the fact that the congestion window is only updated each RTT and not each RTT_0.

B. Simulation 1b

The setup of simulation 1b is like simulation 1a, except that the speed of the UE is increased to 30 m/s (reflecting a fast-moving vehicle). With this modification, we aim to evaluate the Congestion Control algorithms when the amount of time the channel remains in NLOS is much shorter (less than 0.5 second).

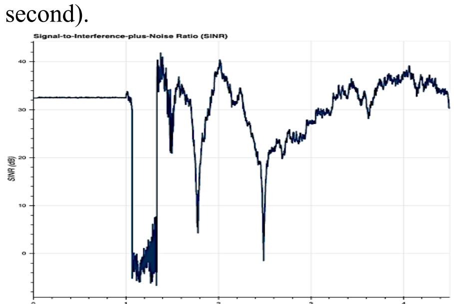


Figure 6: Signal-to-Interference-plus-Noise Ratio observed in Simulation 1b

In this simulation scenario, the UE stands in one place for the first one second (LOS channel condition) and then it enters NLOS at around 1.1 seconds (due to the presence of a building obstacle). We can observe (in figure 6) that the channel exits NLOS very quickly and SINR becomes positive.

Except for TCP New Reno, it takes the same amount of time as simulation 1a (3s) for all other CC algorithms to achieve the maximum throughput (1,000 Mbps) as shown in figure 7. Even when there is an obstacle for a very short period (~ 0.4 s), at least 3 seconds were required to fully utilize the available bandwidth. As a result, when there are multiple obstacles, the throughput is expected to remain low during the entire simulation.

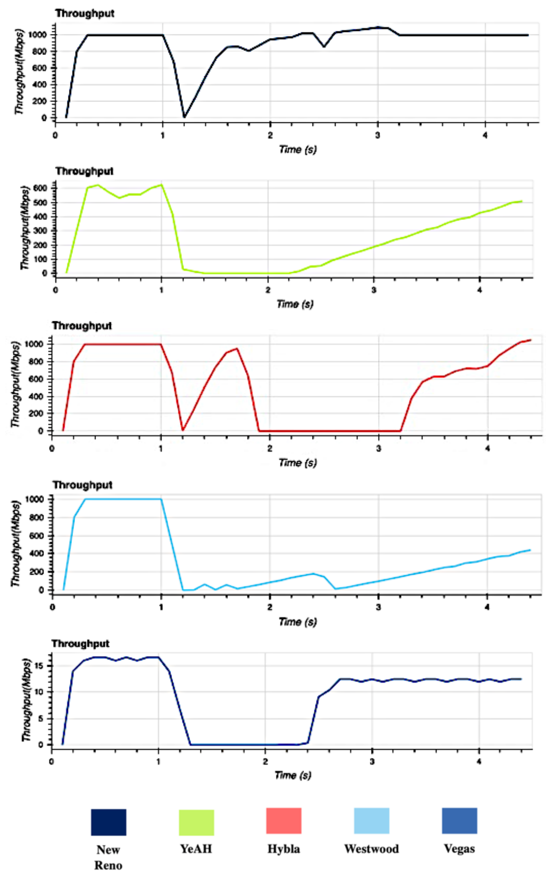


Figure 7: Uplink throughput observed in Simulation 1b

C. Simulation 1c

The setup of simulation 1c (figure 8) also represents a *UMi-StreetCanyon* scenario (like the previous two simulations) but includes a higher number of obstacles (smaller in size) between the UE and the mmWave eNB (all other parameters remain same as Simulation 1a). The speed of the UE is slow (like 1a). This simulation demonstrates the scenario where the channel continuously switches between LOS and NLOS conditions, resulting in the frequent updating of the congestion window size *(cwnd).*

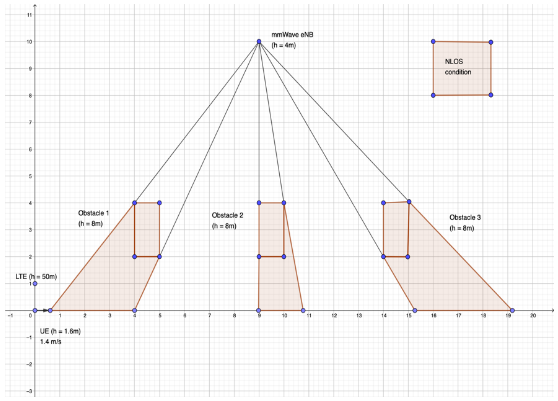


Figure 8: Positions of Network Components in simulation 1c

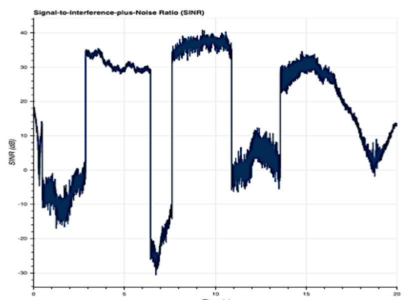


Figure 9: Signal-to-Interference-plus-Noise Ratio (SINR) observed in Simulation 1c

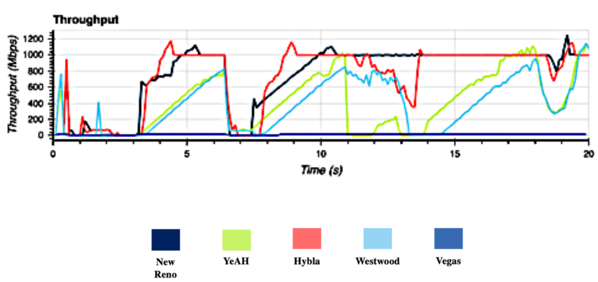


Figure 10: Uplink throughput observed in Simulation 1c

The channel enters NLOS condition three times throughout the period: 0.5 – 3 seconds, 6.4 – 7.7 seconds, and 10.8 – 13.7 seconds. These periods can be seen in figure 9 where the value of SINR becomes negative. Like our results in Simulation 1a, where the UE speed remains low (1.4 m/s), the throughput produced by this simulation using TCP Hybla (figure 10) also took the least time to go up to 1000 Mbps after exiting each NLOS period.

V. Conclusions

In this work we compared the performance of loss-based, delay-based, as well as hybrid TCP Congestion Control algorithms over mmWave communication links. Loss-based algorithms achieved higher throughput than the other algorithms in the absence of any packet losses (typically during Line-of-Sight (LOS) communication). Hybla shows better performance than Westwood or New Reno in terms of congestion window growth rate (and consequentially a higher throughput) when the Non-Line-Of-Sight (NLOS) period ends, and the UE enters the Line-of-Sight (LOS) communication. We hope that the inferences we derive from this study will provide ideas for proposing modifications to existing TCP algorithms to optimize their performances for mmWave networks. Although there exist TCP versions that can handle extremely high bandwidth, more efficient congestion control algorithms are needed which can better utilize the bandwidth of mmWave networks by responding appropriately to signal degradations caused due to high penetration loss and poor isotropic propagation.

In this work the mmWave network is setup by utilizing various models within NS-3 such as the propagation loss model, 3GPP [8] channel model, and mobility models. Each model requires a detailed set of parameter configurations to generate a realistic network model. In future, it will be interesting to repeat this study with the NYUSIM model that has been developed using actual measurements from New York City.

Acknowledgment

This work was supported by the Department of Mathematics and Computer Science at Dickinson College, USA. We are grateful to Tracy McKay and Grant Braught for their inputs that helped in improving the quality and presentation of this research.

References

[1] Huang, K. & Wang, Z. (2011). Millimeter Wave Communication Systems. John Wiley & Sons. pp. Sections 1.1.1–1.2.

[2] Zhang, Ryu, Subramanian, Sampath. (2015). Coverage and Channel Characteristics of Millimeter Wave Band Using Ray Tracing. IEEE International Conference on Communications (ICC).

[3] Turkovic, B., Kuipers, A. F. & Uhlig, S. (2019). Fifty Shades of Congestion Control: A Performance and Interactions Evaluation. *1903.03852. arXiv. cs.NI.*

[4] Caini, C., Firrincieli, R., 2004. (2004). TCP Hybla: a TCP enhancement for heterogeneous networks. International Journal of Satellite Communications and Networking.

[5] Rangan, S., Rappaport, T. S., & Erkip, E. (2014). Millimeter-Wave Cellular Wireless Networks: Potentials and Challenges. Proceedings of the IEEE, vol. 102, no. 3, pp. 366-385.

[6] Zhang, Menglei et al. (2017). Ns-3 Implementation of the 3GPP MIMO Channel Model for Frequency Spectrum Above 6 GHz. Proceedings of the Workshop on ns-3 - 2017 WNS3.

[7] 3GPP. (2017). TR 38.912, Study on New Radio (NR) access technology. V14.0.0. Release 14 (2017- 03), 3rd Generation Partnership Project.

[8] 3GPP. (2017). TR 38.912, Study on New Radio (NR) access technology. V14.0.0. Release 14 (2017- 03), 3rd Generation Partnership Project.

[9] Faria, M., Santos, A., Santos, G., & Spadoti, H. (2019). A System to improve the management of 5G and IoT Networks by determining the Mobile Position. Journal of Microwaves, Optoelectronics and Electromagnetic Applications, 18(2), 293-305.

[10] Henderson, T. R., Lacage, M., Riley, G. F., Dowell, C. & Kopena, J. (2018). Network simulations with the ns-3 simulator. SIGCOMM Demonstration, vol. 14, no. 14, p. 527.