Proportional Fair Spectrum Sharing in a WiFi Network of Co-Existing WiFi 7 and Legacy Nodes

by

Puneet Kumar
(MT23067)

Under the Supervision of Dr **Mukulika Maity**

Indraprastha Institute of Information Technology Delhi
May, 2025

Proportional Fair Spectrum Sharing in a WiFi Network of Co-Existing WiFi 7 and Legacy Nodes

by

Puneet Kumar
(MT23067)

Submitted

in partial fulfillment of the requirements for the degree of

Master of Technology

to

Indraprastha Institute of Information Technology Delhi

May, 2025

Certificate

This is to certify that the thesis titled **Proportional Fair Spectrum Sharing in a WiFi Network of Co-Existing WiFi 7 and Legacy Nodes**, being submitted by **Puneet Kumar**, to the Indraprastha Institute of Information Technology Delhi, for the award of the **Master of Technology**, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other institute or university for the award of any degree or diploma.

May, 2025

M. Maity.

Mukulika Maity
Department of CSE
Indraprastha Institute Of Information Technology Delhi
Delhi 110020

Acknowledgements

Completion of this thesis has been possible only through the generosity, patience, and insight of several individuals, to whom I express my gratitude.

First and foremost, I would like to thank my advisor, Professor Mukulika Maity, for her support, critical feedback, and intellectual generosity. Her thoughtful questions and calm guidance have not only shaped this thesis but also helped me grow into a more independent thinker.

I am particularly grateful to my colleague Jagrati Kulshrestha, whose collaboration and candid conversations brought clarity during moments of uncertainty.

I would also like to acknowledge the guidance provided by Professor Sumit Roy, University of Washington Seattle. Although often challenging, his feedback had critical insights and helped me refine my ideas, and improve the quality of this research. I thank Dr. Hao Yin, University of Washington Seattle, for his feedback in the course of this project.

I thank Prof. Thomas R Henderson, lead maintainer of *ns-3*, for his guidance while implementing WifiCoTraceHelper in *ns-3*. I also thank other *ns-3* developers for their useful review comments. Finally, those unnamed but not forgotten, your support made a quiet but meaningful difference as well.

Puneet

Puneet Kumar

MT23067

May 2025

Abstract

KEYWORDS: IEEE802.11be; WiFi7; Multi-Link Operation; MLO Scheduler; Legacy Devices; *ns-3*; Channel Occupancy

Multi-Link Operation (MLO) in WiFi 7 promises to improve throughput and reduce latency for MLDs (multi-link devices). However, the performance of upcoming WiFi networks where the MLDs (Multi-Link Devices) and legacy SLDs (Single Link Devices) will coexist, is largely unexplored. In this work, we explore such coexistence networks; our initial studies indicate that MLDs achieve disproportionately high throughput as compared to SLDs under traditional DCF-based access. This work is thus primarily directed towards achieving proportional fair throughput allocation to all devices by maximizing a weighted log utility objective function. We leverage the adaptability offered by MLO scheduler in WiFi 7 to meet our objective. Our approach involves designing an MLO scheduler that distributes MLD traffic across multiple links to ensure fairness. We propose two ways in which the MLO scheduler can determine traffic splitting parameters: in one, the access point computes and advertises the parameters to stations, and in the other, each station computes its own parameters using a feedback-based distributed algorithm. We introduce channel occupancy metric as feedback for the distributed algorithm. We implemented both approaches in *ns-3* and contributed features to *ns-3* 3.44 version to measure channel occupancy. Through simulations in *ns-3*, we show that both MLO scheduling approaches can achieve an acceptable level of fairness.

Contents

List of Tables

List of Figures

ABBREVIATIONS

DCF	Distributed Coordination Function
HOL	Head-of-Line
LMAC	Lower MAC
MAC	Medium Access Control
MCAB	Multi-Link Congestion-aware Load Balancing
MLD	Multi-Link Device
MLO	Multi-Link Operation
NSTR	Non-Simultaneous Transmit Receive
NUM	Network Utility Maximization
PMF	Probability Mass Function
SLCI	Single Link Less Congested Interface
SLD	Single-Link Device
STR	Simultaneous Transmit Receive
UMAC	Upper MAC

Chapter 1

Introduction

1.1 Introduction

To meet the increasing demands of high throughput and low latency for emerging applications like AR/VR, etc., IEEE introduced the 802.11be (WiFi7) standard with MLO (Multi-Link Operation) as one of the prominent features to meet these objectives. With MLO, MLDs (Multi-Link Devices) can transmit or receive at the same time on multiple links, either on the same band or on different bands. If the transmission or reception on a link is independent of the other links, it is called STR (Simultaneous Transmit Receive); if dependent on the other links it is NSTR (Non Simultaneous Transmit Receive) mode. In the case of NSTR mode, the transmission on the different links needs to be synchronized whereas in case of STR, the transmissions need not be synchronized. STR mode is more common due to its simplicity of operation and is illustrated in Fig. 1.1 for $L = 2$ links. In this work, we consider only STR-mode of MLO in MLDs.

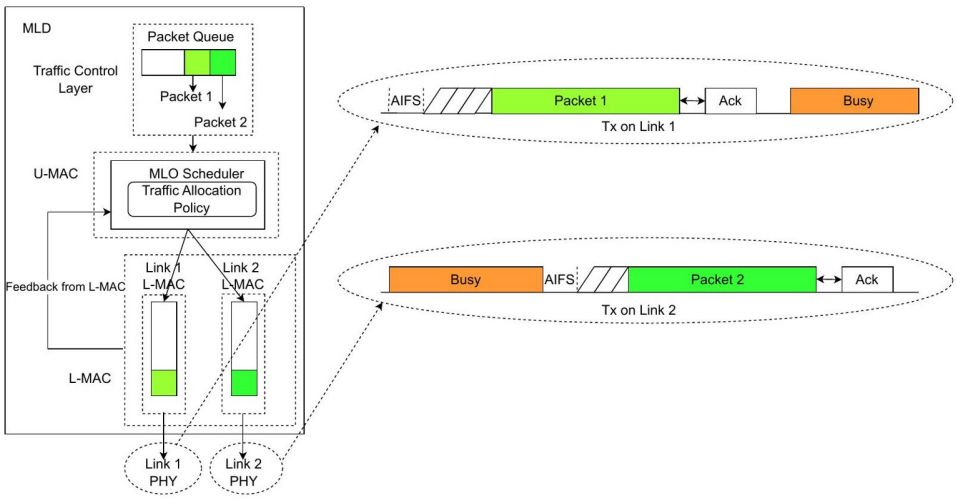


Figure 1.1: Packet transmissions on two links with STR mode.

MLO operation can significantly enhance the performance of MLDs in terms of meeting throughput and/or latency targets [5, 2].

With MLDs in WiFi 7 (and beyond) networks, their coexistence with legacy Single Link Devices (SLDs) is studied in [25]. WiFi networks are backward compatible by design, that presents a trade-off. While it allows legacy devices to operate, it can hinder achieving optimal performance in coexistence. Fig. 1.2 shows a deployment where SLDs and MLDs co-exist on two links on 2.4 GHz and 5 GHz frequency bands. Nodes labeled A, B are SLDs operating on a single link in 2.4 GHz, 5GHz frequency band respectively, whereas node labeled C is a MLD operating on two links in 2.5 GHz & 5GHz bands.

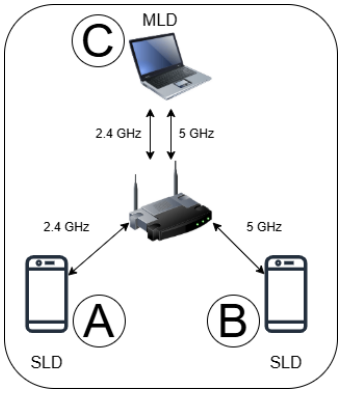


Figure 1.2: Co-existing Network of WiFi 7 MLDs (Multi Link Devices) and legacy SLDs (Single Link Devices).

1.2 Fairness in Co-existing SLD-MLD Networks

The 802.11 DCF procedure aims at providing access fairness over a single link [16]. When SLDs coexist with MLDs, the same DCF procedure at both SLDs and MLDs will lead to throughput fairness over a single link. However, as the MLDs operate on multiple links, they will get an equal access on each of the links. Thus, MLDs obtain a much higher share of aggregate network throughput compared to SLDs. This is also shown through *ns-3* simulations in chapter 3 where it is observed that MLDs on $L = 2$ links can grab twice or more throughput compared to SLDs.

1.3 Overview of Our Work

In this work, we propose solutions to allocate aggregate network throughput in proportional fairness to all devices in a WiFi network with co-existing SLDs and MLDs. A

brief summary of our work and an overview of rest of the chapters is as follows:

1. In chapter 2, we provide background and list related works on NUM, proportional fairness, MLO schedulers etc.

2. In chapter 3, we establish through simulations in *ns-3* that MLDs obtain higher throughput than SLDs in a co-existing WiFi network. The simulation results are used as benchmark against which the proposed solutions are evaluated later.

3. In chapter 4, we use Network Utility Maximization (NUM) framework to model proportional fair throughput allocation problem in a WiFi network with co-existing SLDs and MLDs. We use this model in next chapters to back two practical solutions.

4. In chapter 5, we propose Centralized MLO approach to solve the problem in practice using the NUM model from chapter 4 and a MLO scheduling algorithm.

5. In chapter 6, we propose Decentralized MLO approach that uses a distributed primal control algorithm [22] based on the NUM model from chapter 4.

6. Finally in chapter 7, we mention our contribution to open-source *ns-3* simulation platform and showcase its usefulness in designing channel occupancy-aware MLO schedulers [13, 12].

All of our studies are validated through simulations in *ns-3* and our simulation code is open-source, publicly available at [9, 10].

Chapter 2

Background And Related Work

Fairness is an important consideration in network operations with the coexistence of different types of devices. Multiple definitions of fair sharing of network resources are well-known, such as access fairness (equal number of channel access per node) [16, 3], (air)time fairness (equal successful transmission duration per node) [11, 7], or throughput fairness (equal per-user throughput) [16, 24]. All metrics are equivalent in DCF access when the nodes transmit at the same rate. In this work, we consider throughput fairness only.

2.1 MLO Scheduler

MLO scheduler is a new feature in WiFi 7 standard for multi-link devices and its functionality is to route packets to links. In MLDs, MAC layer is bifurcated into an upper MAC(UMAC) and a lower MAC(LMAC) layers. UMAC has a single queue to buffer incoming packets and LMAC has multiple packet buffer queues, one corresponding to each link. MLO scheduler routes packets to links by dequeuing a packet from the UMAC queue and then enqueuing it to a LMAC queue corresponding to a desired link. Figure 2.1 shows how a MLO scheduler route packets from a single UMAC queue to two LMAC queues on a MLD with two links.

In literature, there are multiple proposals [13, 12] for scheduling algorithms such as a greedy scheduler that schedules a head-of-line (HOL) packet from UMAC to the first link that wins CSMA/CA contention. There are other scheduling algorithms that observe link congestion on each link and then allocates packets to links in some ratio based on historical link congestion. In this work, we will design MLO scheduling algorithms to achieve proportional fairness in WiFi networks.

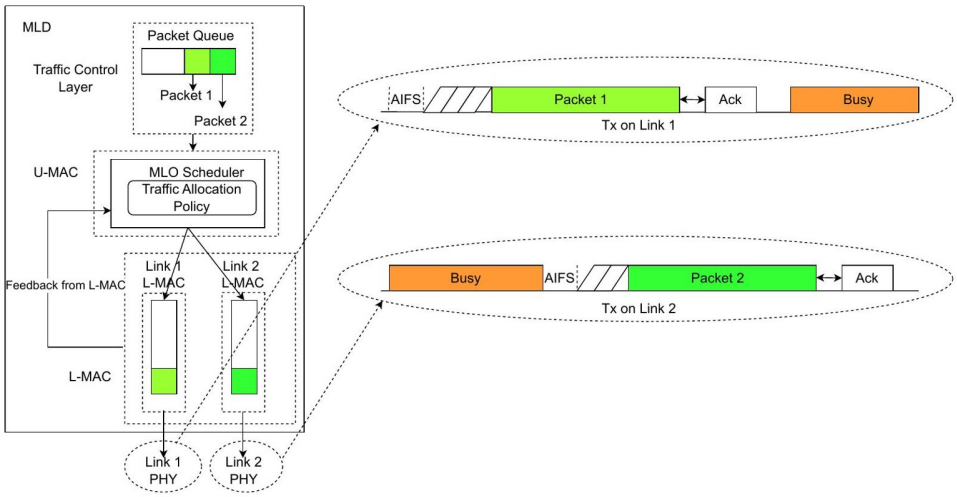


Figure 2.1: MLD Network stack: MLO Scheduler distributes packets from the UMAC packet queue to LMAC packet queues.

2.2 Related Work

[22, 23] are books that describe Network Utility Maximization (NUM) framework to model fair resource allocation problems using abstract concepts like nodes, links to represent a generic network. They also describe different notions of fairness like proportional fairness, max-min fairness etc and their corresponding utility functions to use in NUM formulation. We have used NUM framework to model proportional fair resource allocation in a special case of WiFi 7 network with co-existing MLD and legacy (SLD) nodes. The books [22, 23] describe how to formulate resource allocation problem analytically but they do not offer how to implement a fair resource allocation policy in practical settings like a WiFi network, since it is impossible to cover large variety of networks deployed in practice. In this work, we present two practical approaches [5,6] backed by NUM model to implement proportional fair throughput allocation in WiFi 7 networks, for which we use a new feature called MLO scheduler introduced in WiFi 7.

With MLO, the MAC scheduler must decide how to allocate (application-layer) packets to the links. Prior works [12, 21] have mostly assumed *Greedy Scheduler* as the default, whereby any HOL (Head-of-Line) packet will be transmitted on whichever link first wins the contention. In general, there are multiple other MLO scheduling schemes such as SLCI, MLSA, MCAA [13], MCAB [12], LFTA [8], MH-RSAC [20] that represent the traffic allocation policy. More advanced MLO scheduling policies make use of link-state feedback (i.e. current and recent link-specific information), such

as historical link-congestion, so as to either improve throughput drop ratio or reduce latency. In contrast, we use MLO scheduler for an entirely different purpose of allocating aggregate network throughput in proportional fairness to all nodes. Further, none of these papers consider the co-existence of SLDs and MLDs.

All of the MLO scheduler proposals in [13, 12, 8, 20] were evaluated under different independent simulators. We contributed to the popular open-source *ns-3* simulation platform to measure channel occupancy and evaluated our approaches through *ns-3*. Shen *et al.*[21] studied the impact on latency of a random splitting scheduler compared to a greedy scheduler in *ns-3*. We have used couple of techniques from this paper, such as Random-splitting MLO scheduler and tid-to-link mapping workaround to simulate a MLO scheduler, but our objective and rest of the procedures are entirely different from them. They searched for an optimal split ratio by comparing simulation results over a parameter space with an objective to obtain minimum latency, whereas we find an optimal split ratio to achieve proportional fairness by solving a convex optimization problem formulated using NUM.

Another mechanism to achieve fairness in heterogeneous networks [15, 7, 16, 24, 17, 14, 1] is to optimize DCF-parameters such as contention window parameters for different node types. However, we preferred to use MLO scheduler in this work instead of optimizing contention window parameters because usually hardware vendors do not support tweaking contention window parameters dynamically in devices. On the other hand, MLO scheduler is a new feature in WiFi 7 and we are optimistic that it will be customizable in new WiFi 7 devices.

Chapter 3

Motivation Experiment

3.1 Introduction

In this chapter, we present simulation results from WiFi 7 networks with co-existing MLDs and legacy devices (SLDs). We vary the number of SLDs and MLDs on two links of the network and observe throughput obtained by SLDs and MLDs. These results will act as benchmark for evaluating improvement in later chapters.

3.2 Motivation

With the introduction of WiFi 7 standard, it is inevitable that all legacy devices will be replaced by the new standard eventually. But this transition to the new standard will take at least a few years, and legacy and WiFi 7 standard devices will co-exist during the transition. Our motive is to study the impact of introducing WiFi 7 MLD on legacy SLD in a co-existing WiFi network. We conduct simulations in *ns-3* that supports MLO since version *v3.42* for this study.

3.3 Setup

We conduct a few initial benchmarking simulations based on the per-STA throughput of both SLDs and MLDs in a coexistence scenario. We evaluate the throughput for a saturated BE network with uplink-only traffic. We compute the per-STA throughput through simulations in *ns-3*. We assume STR mode for MLDs with $L = 2$ links. Further, a *greedy scheduler* [21] is used, which transmits the packet(s) on the link that wins the channel access first. The other simulation parameters are given in Table 3.1.

We keep the simulation parameters simple and symmetric for all devices to keep analysis simple and focused on the inherent difference between MLD and SLD, i.e. num-

Table 3.1: Simulation parameters

Parameter	Value	Parameter	Value
Radius	0.1m	Payload Size	1000 B
Link 1	2.4 GHz	Guard Interval	800 ns
Link 2	5 GHz	MCS	11
MSDU Aggregation	Disabled	MPDU Aggregation	Disabled
Bandwidth	40MHz on each link	Simulation Duration	10-35 sec

ber of links on which they operate. We use following notation to describe a simulation scenario:

Table 3.2: Notation for simulation configurations.

Parameter	Description
nSld1	Number of SLDs on Link 1
nSld2	Number of SLDs on Link 2
nMld	Number of MLDs. MLDs operate on both links 1 and 2.

3.4 Simulation Results

In the following graphs shown in figures 3.1-3.4 for different simulation configurations, we plot a metric called Throughput-Ratio on each link. The metric Throughput-Ratio is defined as:

$$\text{Throughput-Ratio} = \frac{\text{Average MLD total throughput on both links}}{\text{Average SLD throughput on a single link}} \tag{3.1}$$

Since the simulation parameters are symmetrical for devices, we obtain almost equal throughput in each of the three categories of device, i.e. MLDs, SLDs on link 1 and SLDs on link 2. We took average of throughputs in each category to compute the metric Throughput-Ratio (3.1).

Going forward, we label the benchmarking results obtained in this chapter as 'Default' in graphs. The simulation configuration for each figure is mentioned in their captions using the notation introduced previously in table 3.2.

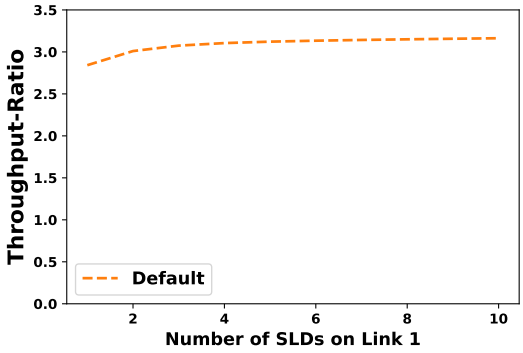
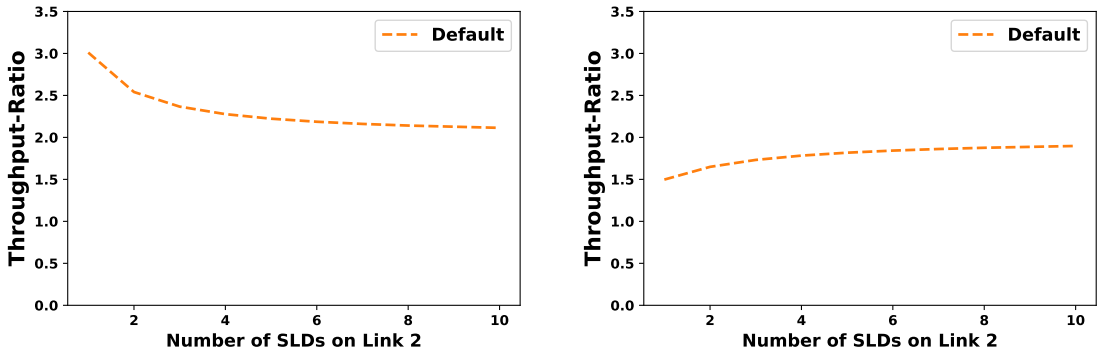
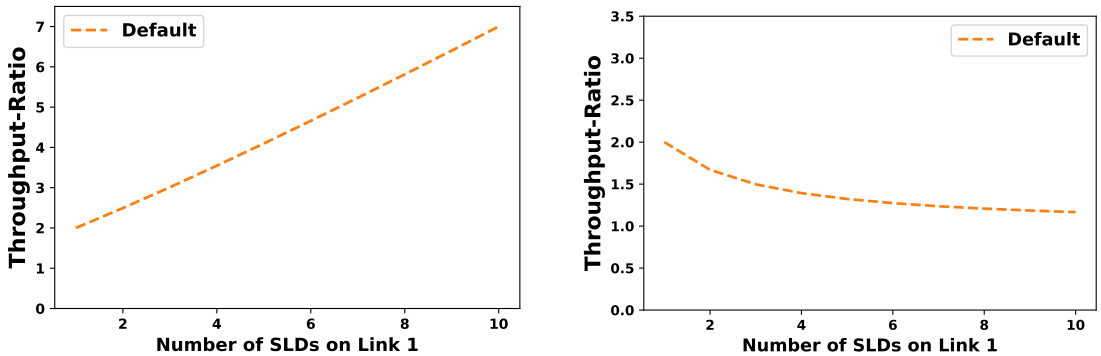


Figure 3.1: Simulation Configuration: nSld1 = nMld, nSld2 = 0



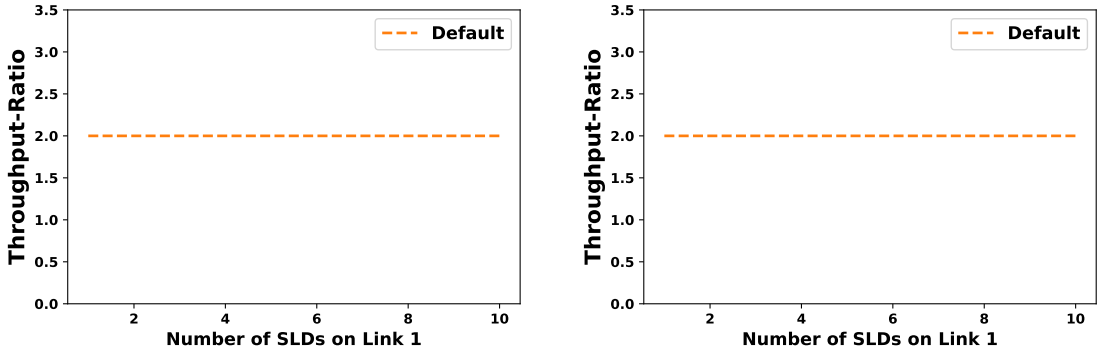
(a) Link 1 (b) Link 2

Figure 3.2: Simulation Configuration: nMld = nSld2 = nSld1 - 2



(a) Link 1 (b) Link 2

Figure 3.3: Simulation Configuration: nMld = nSld2 = 1



(a) Link 1 (b) Link 2

Figure 3.4: Simulation Configuration: nSld1 = nSld2 = nMld

3.5 Analysis

We observe in figures 3.1-3.4 that MLDs obtain 2-3x times the throughput than SLDs in majority of cases. Further, the difference in Throughput-Ratio depends on network configuration, i.e. number of MLDs and SLDs on each link.

3.6 Conclusion

We establish through simulations that MLDs obtain higher throughput than SLDs in a co-existing WiFi network. We presented different simulation scenarios along with a concise notation to denote them, which will be reused in later chapters. The current simulation results will act as benchmark against which the proposed solutions will be evaluated later.

Chapter 4

Problem Formulation

4.1 Introduction

In chapter 3, we showed that MLDs obtain higher throughput compared to SLDs when they co-exist in a WiFi 7 network. In this chapter, we intend to control the gap between throughputs of MLDs and SLDs, and for that we formulate the problem as maximization of an objective function. We view a link's capacity as a resource, and we allocate that resource (throughput) to each device using Network Utility Maximization (NUM) framework. Then we compare the throughput ratio obtained from maximizing the objective function versus the default behavior of greedy MLO scheduler.

4.2 Network Utility Maximization (NUM) Problem

4.2.1 Objective Function

We intend to allocate throughput to MLDs and SLDs such that MLD-to-SLD throughput ratio (3.1) is $\gamma : 1$. In case when γ equals 1, throughput is allocated equally to both MLDs and SLDs. γ can be greater than or equal to 1 but less than the value obtained through greedy scheduler.

The optimization problem specified in equation (4.1) is the objective function and symbols used in the equation are defined in table 4.1.

$$\max_{Th_i^{\phi,l}} \sum_{i=1}^{n} \left(w_i * \log(Th_i^{\Phi})\right) \tag{4.1}$$

$$Th_i^{\Phi} = \sum_{l=1}^{\eta} Th_i^{\Phi,l} \quad if\ \Phi = MLD$$

with the following constraints:

$$\sum_{i=1}^{n} Th_i^{\Phi,l} \leq Sat^l \quad \forall l$$

$$Th_i^{\Phi,l} \geq 0 \quad \forall \Phi, l, i$$

$$w_i = 1 \quad if\ \Phi = SLD$$

$$w_i = \gamma \quad if\ \Phi = MLD$$

Table 4.1: List of Symbols.

Symbol	Description
$Th_i^{\Phi,l}$	throughput of a device i on link l. Device type is Φ.
Th_i^{Φ}	total throughput of a device i on all links. Device type is Φ.
w_i	weight assigned to device i.
Φ	Device type, either MLD or SLD.
Sat^l	saturation throughput achievable on link l.
l	Link
η	No. of links
n	Total No. of devices. Only STAs are included, AP is excluded.
γ	A positive constant ≥ 1.

4.2.2 Interpretation of Objective Function as NUM Problem

Network utility maximization (NUM) framework is a mathematical model of the problem of fairly allocating resources among users[23]. In the context of our problem, we consider a link's capacity, i.e. Sat^l, as a resource that is to be allocated among MLDs and SLDs of the network fairly. $Th_i^{\phi,l}$ is the throughput allocated to node i on link l and Th_i^{ϕ} is the total throughput allocated to node i on all links. In NUM framework, each user has an associated utility function to quantify utility of resources allocated to it, where a utility function is a mathematical function that maps resource allocation to a real number. We used weighted log utility function ($w_i \log$) in equation (4.1) as utility function of node i, the reason for this choice is described shortly. Hence, the objective function can be interpreted as maximizing the aggregate network utility, i.e. sum of utility of each node of the network, under the constraints that aggregate throughput on any link is within the link's capacity.

There are different notions of fairness and weighted log utility provides proportional fairness[23]. Log function has a property of diminishing marginal utility, i.e. the

additional utility obtained by assigning more resources to a node keeps diminishing as shown in figure 4.1. So, in order to maximize aggregate network utility, the throughput allocations Th_i^{ϕ} has to be fair.

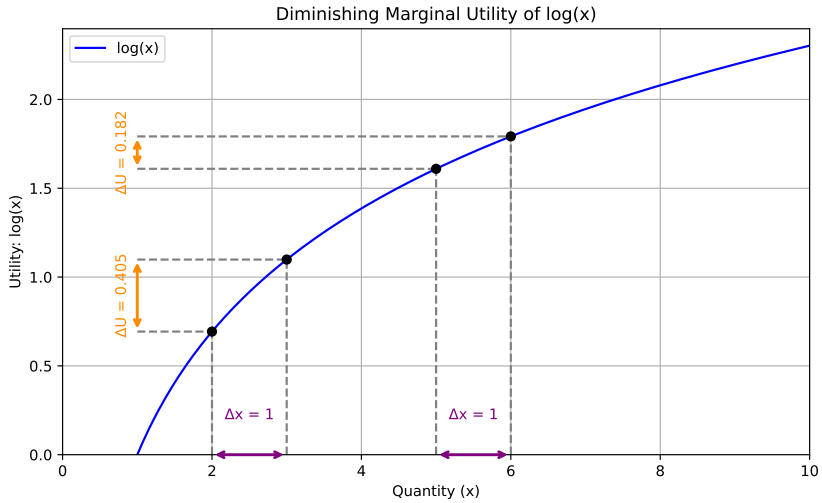


Figure 4.1: Diminishing Marginal Utility of Log function.

An important aspect of the objective function in equation (4.1) is that it is *Pareto efficient* because it does not waste link capacity, even if it leads to much higher throughput ratio than $\gamma : 1$. For example, if there are no SLDs on link 2 then the objective function (4.1) allocates entire capacity of link 2 to MLD in order to maximize the network utility, it does not matter how high the throughput ratio becomes. For this same example of no SLD on link 2, some other objective function could choose to not allocate the entire capacity of link 2 to MLD just for the sake of maintaining the throughput ratio to $\gamma : 1$, but then that other objective function would be wasting capacity and would not be called as Pareto efficient because it would be possible to increase MLD's throughput using the leftover link capacity without the need to take it from any SLD' share of throughput.

Moreover, the optimization problem (4.1) is a convex optimization problem and can be solved with convex solvers like CVXPY [6] efficiently.

4.3 Comparison of Proportional Fair and Default Behavior

We solve equation (4.1) using a convex solver like CVXPY [6] for different simulation scenarios in motivation experiments listed in chapter 3. Going forward, we label the

solutions to equation (4.1) as proportional fair (Prop-Fair) in graphs. In the following graphs shown in figures 4.2-4.6 for different simulation configurations, we compare the proportional fair throughput allocation with the default greedy scheduler. The simulation configuration for each figure is mentioned in their captions using the notation introduced previously in chapter 3. γ is the weight assigned to MLDs in equation (4.1) whereas weight assigned to SLDs is 1. γ is also set equal to 1 for most scenarios except in figure 4.5 where it is set to 2.

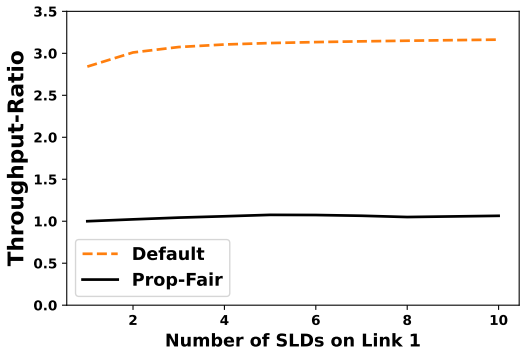
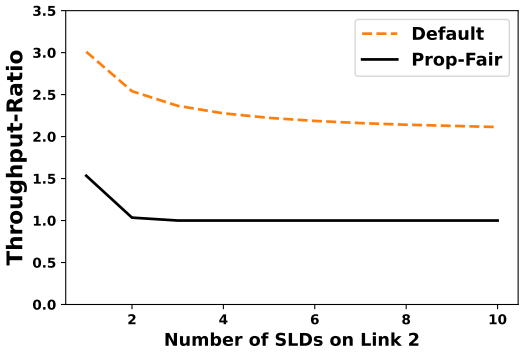
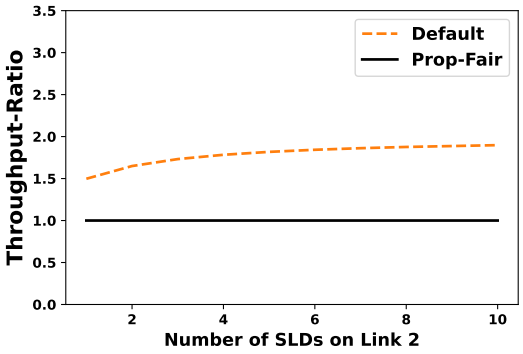


Figure 4.2: Simulation Configuration: nSld1 = nMld, nSld2 = 0

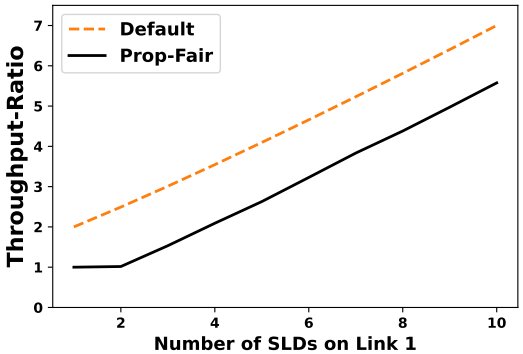


(a) Link 1

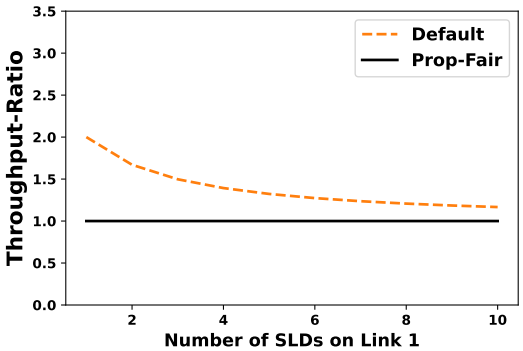


(b) Link 2

Figure 4.3: Simulation Configuration: nMld = nSld2 = nSld1 - 2

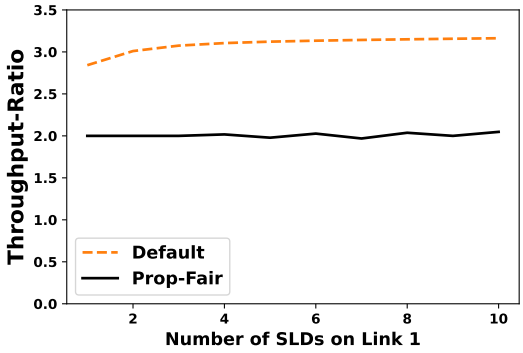


(a) Link 1



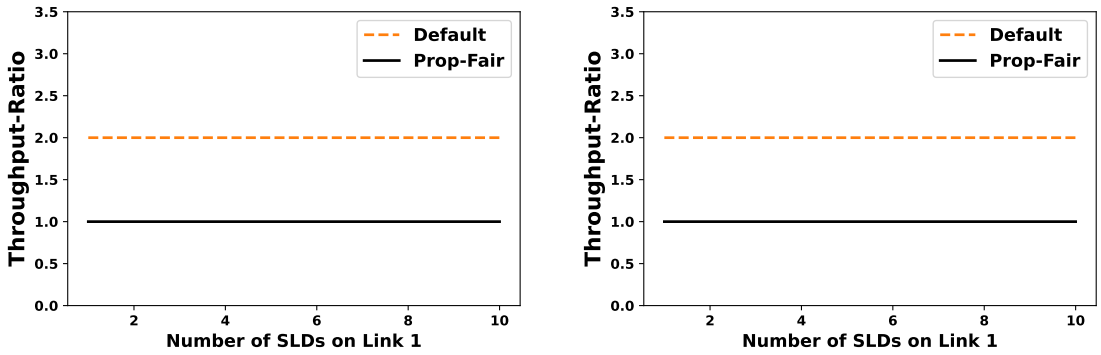
(b) Link 2

Figure 4.4: Simulation Configuration: nMld = nSld2 = 1



(a) Link 1

Figure 4.5: Simulation Configuration: $\gamma = 2$, nSld1 = nMld, nSld2 = 0



(a) Link 1 (b) Link 2

Figure 4.6: Simulation Configuration: nSld1 = nSld2 = nMld

4.3.1 Analysis

The graphs show that it is possible to meet the objective of obtaining throughput ratio of $\gamma : 1$ by adjusting throughput allocation except in figure 4.4. In 4.4, link 1 is highly congested compared to link 2, therefore MLD obtains much higher throughput from link 2 compared to SLDs on link 1. So, "Prop-Fair" in 4.4a shows that we cannot reduce throughput ratio to 1:1, but we can reduce that ratio somewhat.

4.4 Conclusion

In this chapter, we proposed an analytical method to compute proportional fair throughput allocation by solving a convex optimization problem in equation (4.1). In the next two chapters, we will build on the analytical method and propose two implementable approaches to achieve proportional fair throughput allocation in practice.

Chapter 5

Centralized Solution

5.1 Introduction

In this chapter, we propose to utilize MLO scheduler to achieve proportional fair throughput allocation. First, we present a MLO scheduling algorithm, called Random-Splitting, that achieves proportional fairness given that the algorithm is provided the desired throughput allocations, i.e. $Th_i^{\phi,l}$ defined in table 4.1, as input. Then, we propose our first practically implementable approach called Centralized MLO approach that utilizes Random-Splitting MLO scheduling algorithm. We evaluate Centralized MLO approach through simulations in ns-3 and present the results in the end.

5.2 MLO Scheduler

MLO scheduler is described in section 2.1. In literature, there are multiple proposals [13, 12] for scheduling algorithms, such as a greedy scheduler that schedules a head-of-line (HOL) packet from UMAC to the first link that wins CSMA/CA contention, algorithms that observe link congestion on each link and then allocate packets to links in some ratio based on historical link congestion. However, our objective to use MLO scheduler is different from those proposals because they want to reduce packet latencies whereas we want to achieve proportionally fair throughput. We plan to use MLO scheduler such that a MLO scheduler schedules a number of packets on each link required to reach a pre-determined throughput value , i.e. $Th_i^{\phi,l}$, approximately. In the next section we propose a MLO scheduling algorithm to achieve our objective of proportional fairness.

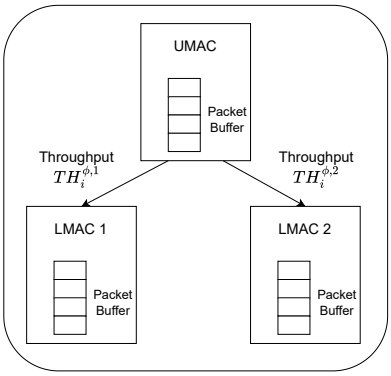


Figure 5.1: MLO Scheduler Design: Dequeue packets from UMAC's packet buffer and enqueue them to LMAC 1 and LMAC 2 at rates $TH_i^{\phi,1}$ and $TH_i^{\phi,2}$ respectively.

5.3 Random-Splitting MLO Scheduling Algorithm

Given that a MLD has a saturated load and assuming that a MLD somehow knows its desired throughput allocation on all links, we design a MLO scheduling algorithm to achieve the desired throughput allocation. An example of the algorithm's objective is shown in figure 5.1 for a MLD with two links.

Algorithm 1 Random-Splitting MLO Scheduling Algorithm
Description: Given $Th_i^{\phi,l}$ for each link l on MLD i, this MLO scheduling algorithm running on MLD i achieves desired throughput allocation $Th_i^{\phi,l}$ on each link l.

Require: $Th_i^{\phi,l}$ = Throughput allocation on each link l of MLD i.

Rate-limit incoming saturated load from UMAC to $\sum_{l=1}^{\eta} Th_i^{\Phi,l}$.

Let X be a discrete random variable with range $S_X = \{1, 2 \ldots, \eta\}$
Define PMF $P_X[X = x] = \frac{Th_i^{\phi,x}}{\sum_{l=1}^{\eta} Th_i^{\Phi,l}}$

for each packet p $\in$ rate-limited load **do**

l = Randomly choose a link l using PMF P_X

Schedule packet p on link l

end for

First, the MLO scheduler rate-limits the incoming saturated load to a value equal

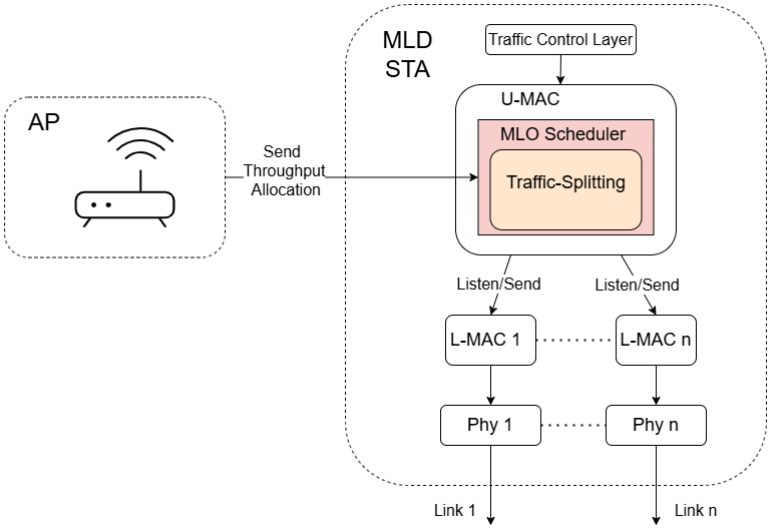


Figure 5.2: Centralized MLO: Access point calculates and distributes throughput allocation to MLDs.

to the total outgoing throughput on all links, i.e. $Th_i^{\phi} = \sum_{l=1}^{\eta} Th_i^{\Phi,l}$. Then, it uses a technique called Random-Splitting, proposed in [21], to split the rate-limited load among all links in a desired split-ratio (5.1).

$$\text{Split-Ratio}_i = Th_i^{\Phi,1} : Th_i^{\Phi,2} : ... : Th_i^{\Phi,\eta} \tag{5.1}$$

The algorithm uses a discrete random variance X which is used to choose a link from 1 to η for each incoming packet. The probability to choose a link x is directly proportional to the desired throughput $Th_i^{\phi,x}$ on that link. The algorithm is specified in Algorithm 1.

5.4 Centralized MLO Approach

Now we tackle the question of how each MLD obtains its throughput allocation $Th_i^{\phi,l}$. We propose that access point(AP) constructs the convex optimization problem in equation (4.1) and solves it using a convex solver and sends the solution to MLDs as shown in figure 5.2. Since the access point is aware of all devices and their mode of operation, whether single-link or multi-link, so the access point has the required information to construct the convex optimization problem. Since we have limited the scope of our problem to a simplified network setting with saturated uplink load, identical

MCS without packet aggregation etc as described in chapter 3, the access point can determine saturated load Sat^l using analytical model for DCF-mechanism proposed in [4].

We highlight the point that we do not touch SLDs and let them function as it is because only MLDs need to limit their throughput, not the SLDs. MLDs sacrifice some share of aggregate throughput which gets equally distributed among SLDs because CSMA/CA gives fair opportunities to transmit.

5.5 Simulation in ns-3

We evaluated Central MLO approach for all simulation scenarios mentioned in chapter 3 through simulations in ns-3. In this section, we mention how we simulated a MLO scheduler in ns-3 and then present the simulation results.

5.5.1 Simulating MLO Scheduler in *ns-3*

We need to replace the default MLO scheduler with a custom implementation of random-splitting MLO scheduler proposed in algorithm 1, but *ns-3* does not provide such a provision. So, we used a workaround mentioned in [21] to simulate our custom MLO scheduler. The workaround is to setup tid-to-link mapping during MLO setup such that tid '0' packets are sent on the first link and tid '3' packets on the second. After the MLO setup, we generate packets at the application layer with the two types of tid, '0' and '3', in a proportion called *split-ratio* defined in equation (5.1). This workaround enables us to implement a custom MLO scheduler by controlling both the rate of traffic generation and the split-ratio at the application layer.

5.5.2 Simulating Centralized MLO Approach in *ns-3*

Random-splitting MLO scheduling algorithm requires throughput allocation $Th_i^{\phi,l}$, so first we solve the convex optimization problem in equation (4.1) using CVXPY convex solver library[6] in Python to obtain throughput allocated to each node on each link ($Th_i^{\phi,l}$). We calculate the saturated throughput (Sat^l), that's required in the constraints

of equation (4.1), using Bianchi's analytical model [4]. Bianchi's analytical model is applicable here because the simulation parameters in Table 3.1 are chosen such that the simulation results can be compared with the analytical ones. After obtaining the throughput allocations from convex solver, we use them in *ns-3* simulation. In the simulation, we let SLDs generate uplink traffic at a saturated load of 35 Mbps but we control the uplink traffic generated at MLDs according to the workaround explained previously in 5.5.1. Let's say a MLD is allocated throughputs $r1$ and $r2$ on the two links, then the MLD generates traffic at $r1 + r2$ and its split-ratio is $r1 : r2$ in the workaround. We run the simulation for 10 seconds and measure the average throughput obtained by each device type by counting the number of packets received at server.

5.5.3 Simulation Results

In the following graphs shown in figures 5.3-5.7 for different simulation configurations, we compare throughput-ratio obtained using centralized MLO approach, labeled as Central-MLO in graphs, with proportional fair ratio. The simulation configuration for each figure is mentioned in their captions using the notation introduced previously in chapter 3.

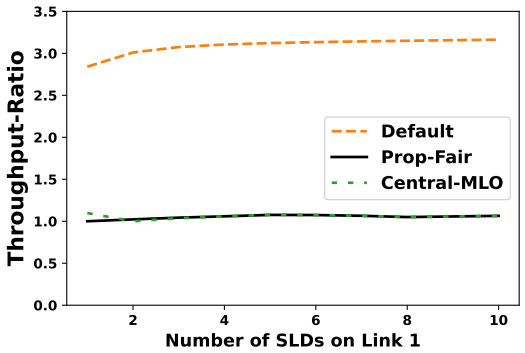
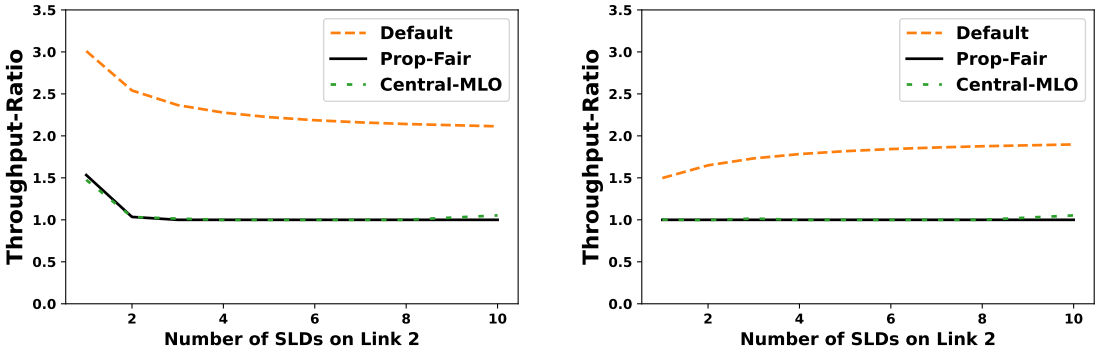
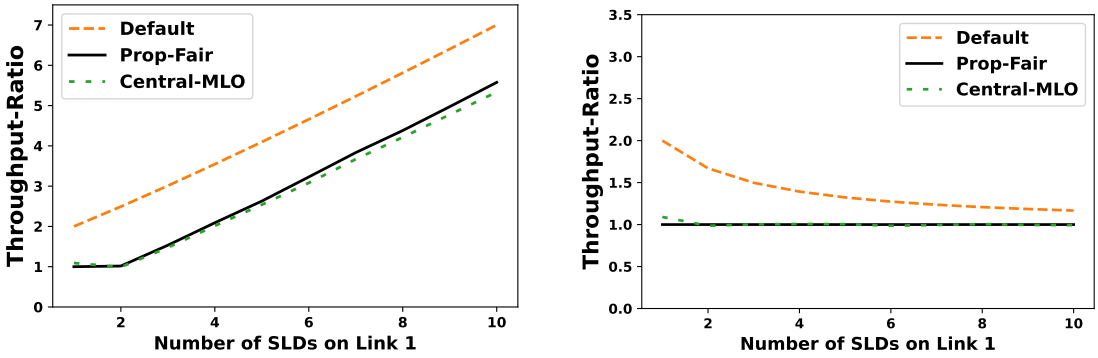


Figure 5.3: Simulation Configuration: nSld1 = nMld, nSld2 = 0



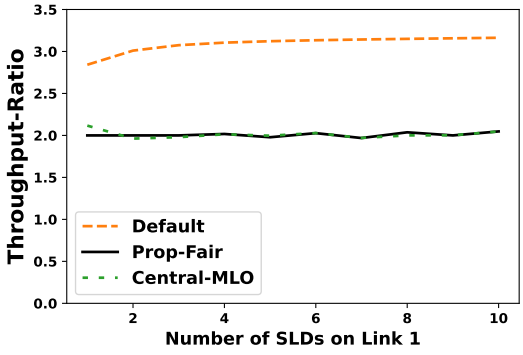
(a) Link 1 (b) Link 2

Figure 5.4: Simulation Configuration: nMld = nSld2 = nSld1 - 2



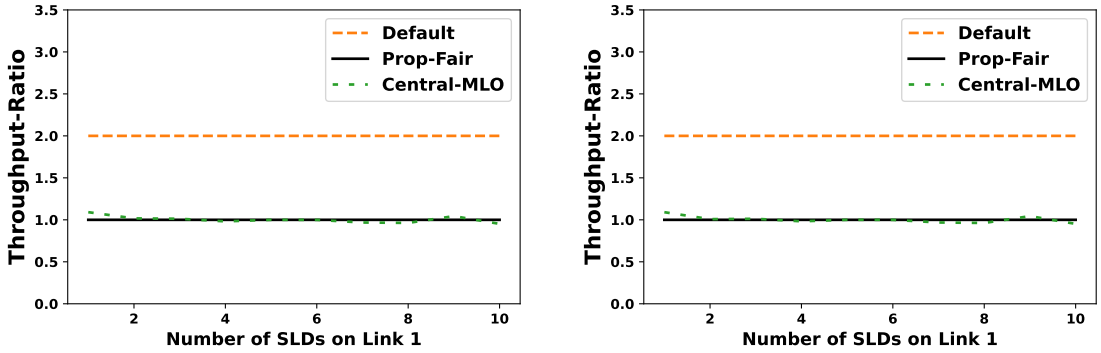
(a) Link 1 (b) Link 2

Figure 5.5: Simulation Configuration: nMld = nSld2 = 1



(a) Link 1

Figure 5.6: Simulation Configuration: $\gamma = 2$, nSld1 = nMld, nSld2 = 0



(a) Link 1 (b) Link 2

Figure 5.7: Simulation Configuration: nSld1 = nSld2 = nMld

5.5.4 Analysis

We observe that centralized MLO approach almost resembles proportional fair throughput allocation in all simulation scenarios. Some minor deviations can be attributed to randomness introduced in random-splitting MLO scheduling algorithm, and other sources of randomness in *ns-3* like packet generation etc.

5.6 Conclusion

The simulation results show that centralized MLO approach meets our objective of proportional fairness very well. In the next chapter, we propose another approach which reuses two ideas described in this chapter: Random-splitting MLO scheduling algorithm 5.3 and simulating MLO scheduler in *ns-3* 5.5.1.

Chapter 6

Decentralized Solution

6.1 Introduction

So far, the solution to achieve proportional fairness is reduced to determining throughput at each link for all devices. In the previous Centralized MLO approach, we determined throughputs by solving a constrained convex optimization problem at the AP and then send the solution from AP to each MLD. But this approach has an overhead for AP which takes an extra responsibility of controlling throughput allocation of the network. In the new Decentralized MLO approach, we use distributed primal control algorithm described in [22]. Using this distributed algorithm, a device can determine fair throughput values on its links on its own by sensing congestion on the links, and hence it eliminates the need of any centralized controller to control throughput allocation. Next, we provide a brief description of distributed primal control algorithm and then tell how we applied it to a co-existing WiFi network. We propose a novel metric called Channel Occupancy that is useful in applying the distributed algorithm in a WiFi network.

6.2 Distributed Primal Control Algorithm

In Eq. (4.1), we solved an optimization problem with constraints. Here, we start with converting Eq. (4.1) to an equivalent unconstrained optimization problem as shown in the following equation:

$$\max_{Th_i^{\Phi,l}} \sum_{i=1}^{n} \left(w_i * \log(Th_i^{\Phi})\right) - \sum_{l=1}^{\eta} P(\sum_{i=1}^{n} Th_i^{\Phi,l}) \tag{6.1}$$

by appending the constraints to the objective function as $\sum_{l=1}^{\eta} P(\sum_{i=1}^{n} Th_i^{\Phi,l})$. The function P in these equations can be interpreted as the price or penalty of using a link as it associates a price to carry the total throughput $\sum_{i=1}^{n} Th_i^{\Phi,l}$ through a link l. Ideally, the penalty function P should approach infinity as $\sum_{i=1}^{n} Th_i^{\Phi,l}$ approach the saturated

load sat^l on link l, and P should be zero elsewhere for Eq. (4.1) and Eq. (6.1) to be equivalent. But such an ideal penalty function P is not differentiable and hence impractical for our purpose, so we will approximate the ideal penalty function with a smooth and steep polynomial function in equation (6.2).

$$P(\sum_{i=1}^{n} Th_i^{\Phi,l}) = \mu * (max(0\ ,\ \sum_{i=1}^{n} Th_i^{\Phi,l} - 0.8 * Sat^l))^2 \tag{6.2}$$

where Sat^l is aggregate throughput on link l at saturation, and μ is a constant to adjust steepness of the penalty function. We choose this penalty function because it increases steeply when co^l approaches sat^l, and is differentiable everywhere which is required to compute gradient in equation (6.3) as we describe it next.

Next, we take gradient of Eq. (6.1) with respect to $Th_i^{\Phi,l}$ and use that gradient value in gradient ascent algorithm to derive the following equation:

$$\dot{Th_i^{\Phi,l}} = S * \left(\frac{w_i}{Th_i^{\Phi}} - P'(\sum_{i=1}^{n} Th_i^{\Phi,l}) \right) \tag{6.3}$$

where $\dot{Th_i^{\Phi,l}}$ is the derivative of $Th_i^{\Phi,l}$ with respect to time.

Eq. (6.3) can be interpreted as a control system where throughput $Th_i^{\Phi,l}$ on a link increases when the device's throughput Th_i^{Φ} is low and the link is less congested, i.e. $\sum_{i=1}^{n} Th_i^{\Phi,l}$ is low as well. However, as the link becomes congested and the device's throughput increases, the rate of increase of throughput $Th_i^{\Phi,l}$ decreases and eventually stabilizes at an equilibrium. [22] mentions that equilibrium point of Eq. (6.3) is globally asymptotic stable and it is also the optimal solution of our objective function in Eq. (6.1) if we choose a convex penalty function P,say a polynomial like in eq (6.2), and set the step-size S equal to $Th_i^{\phi,l}$.

Each device updates its throughput allocation with time as per Eq. (6.3), i.e., at discrete time-intervals a device i computes the delta by which it should update its throughput using the expression on the right-hand side of the equation. We need to apply Eq. (6.3) to update throughput at discrete time-steps, say every 0.5 second, which requires an estimate of total throughput $\sum_{i=1}^{n} Th_i^{\Phi,l}$ flowing through a link l in the latest time-window. The total throughput $\sum_{i=1}^{n} Th_i^{\Phi,l}$ can also be interpreted as a measure of link congestion. [22] mentions some approaches to estimate it in the internet, such

as introducing a new field in a packet header to record throughput as a packet passes through links, or assume a lost packet is a signal that a link is congested. Instead of using these estimation approaches, we propose a novel metric called Channel Occupancy which is suitable to estimate link congestion on WiFi networks and is easily measurable at PHY layer as a by-product of DCF mechanism.

6.3 Channel Occupancy - A Proxy Metric for Link Congestion

We define a novel metric called Channel Occupancy as a proxy to estimate congestion on link, i.e. $\sum_{i=1}^{n} Th_i^{\Phi,l}$ in Eq. (6.3). Channel Occupancy is defined as fraction of time a link is observed to be busy in a time-window and we propose to calculate it at PHY layer during DCF as follows:

$$co = \frac{\text{Time-Window Duration} - \text{Idle Duration}}{\text{Time-window Duration}} \tag{6.4}$$

During DCF, a PHY layer senses its channel and transitions through multiple states such as IDLE, RX, TX etc.

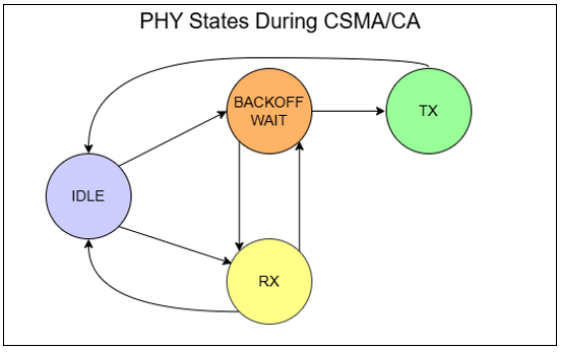


Figure 6.1: A few state-transitions during CSMA/CA protocol.

We measure the time-duration for which a PHY is in IDLE state and use it to compute Channel Occupancy co during a time-window using Eq. (6.4). We assume the PHY goes in a sleep state for negligible time and the channel is assumed to be busy when a PHY is in sleep state. This assumption seems valid when each device has a saturated load and therefore a PHY always has a frame to transmit. Eq. (6.3) changes to the following

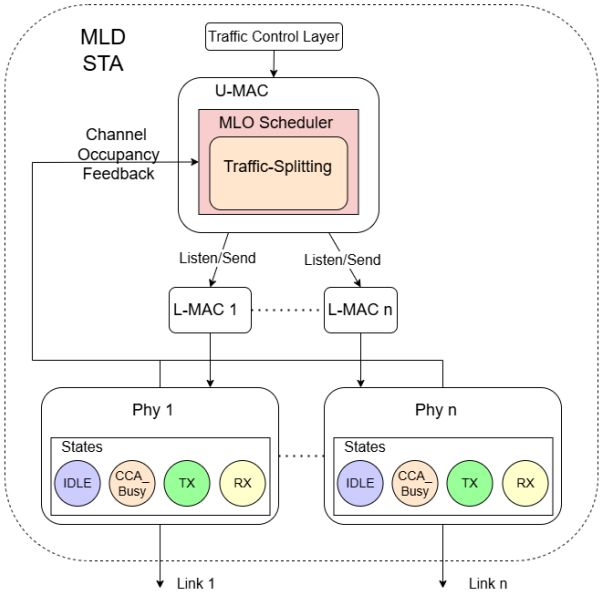


Figure 6.2: In Decentralized MLO, each MLD senses channel occupancy to calculate its throughput allocation.

equation when we replace $\sum_{i=1}^{n} Th_i^{\Phi,l}$ with Channel Occupancy:

$$\dot{Th}_i^{\Phi,l} = S * \left(\frac{w_i}{Th_i^{\Phi}} - P'(co^l) \right) \tag{6.5}$$

where co^l is Channel Occupancy on link l, and Sat^l in P now represents saturated channel occupancy instead of saturated throughput on link l.

In the next section, we describe how distributed primal control algorithm and channel occupancy are applied together in Decentralized MLO approach.

6.4 Decentralized MLO Approach

The primary difference in the decentralized approach compared to the centralized approach is that each device calculates its throughput using a local feedback controller instead of receiving it from a central controller. Each device's local feedback controller continuously measures channel occupancy on its links and uses equation (6.5) to adjust its throughput. Hence, it eliminates the need for AP to assume an extra responsibility of controlling throughput allocation on the network.

The local feedback controller mentioned above is a combination of functionalities at the PHY layer and MLO scheduler. Figure 6.2 shows how Channel Occupancy co^l is

measured on each link during a time-window and then provided as feedback to the MLO scheduler. The MLO scheduler applies Eq. (6.5) to update throughput on each link upon receiving Channel Occupancy at the end of the time-window.

Algorithm 2 Local Feedback Controller Algorithm on Device i
Description: Device i adjusts its throughput allocation $Th_i^{\phi,l}$ using channel occupancy co^l

while true **do**

for each link l **do**

co^l =Calculate channel occupancy on link l using PHY state-transition measurements.

$\Delta Th_i^{\Phi,l} = S * \left(\frac{w_i}{Th_i^{\Phi}} - P'(co^l)\right)$

end for

for each link l **do**

$Th_i^{\Phi,l}$ += $\Delta Th_i^{\Phi,l}$

end for

Sleep for time t, say $t = 0.5$ seconds

end while

In algorithm 2, we calculate the updates $\Delta Th_i^{\Phi,l}$ using PHY state-transition measurements from the latest time-window in the first **for** loop. The point to note is that the updates are calculated using throughput values from the previous time-window according to gradient ascent algorithm, and hence the throughputs are not updated until all updates are calculated. We repeat this process every 't' time-interval, say 0.5 seconds.

Once the throughputs are calculated according to algorithm 2, the calculated throughput values are provided to Random-splitting MLO scheduler algorithm 1, same as in Centralized MLO approach, which adjusts the throughput on each link accordingly. Hence, Decentralized MLO approach differs from Centralized MLO approach only in the way a device obtains its throughput allocation.

6.5 Simulation in *ns-3*

We evaluated Decentralized MLO approach for all simulation scenarios mentioned in chapter 3 through simulations in *ns-3*. In this section, we mention how we simulated the

local feedback controller algorithm 2 in *ns-3* and then present the simulation results.

6.5.1 Simulating Decentralized MLO in *ns-3*

In these simulations, we start with generating load at a low rate of 1 Mbps on each node and use the same simulation parameters from Table 3.1. We simulate local feedback controllers, each one functioning in isolation on a node, that measure channel occupancy and update their node's throughput periodically according to equation (6.5). We describe how we measure channel occupancy in *ns-3* in chapter 7. In the equation, parameter S is set to $Th_i^{\phi,l}$ and parameter μ to $\frac{1}{Sat^l}$ for all simulation scenarios. While the value for parameter S is suggested in [22], we determined the value for parameter μ through trial-and-error in simulations. The feedback controller measures channel occupancy from a node's viewpoint using an analytically validated and reusable helper class that we contributed to ns-3, its details are in next chapter 7. The feedback controllers also use the same workaround mentioned previously in section 5.5.1 to split traffic between links on MLDs. We run the simulation for 35 seconds and measure the average throughput in the last 10 seconds of the simulation.

6.5.2 Simulation Results

In the following graphs shown in figures 6.3-6.7 for different simulation configurations, we compare throughput-ratio obtained using Decentralized MLO approach, labeled as Distrib-MLO in graphs, with proportional fair ratio. The simulation configuration for each figure is mentioned in their captions using the notation introduced previously in chapter 3.

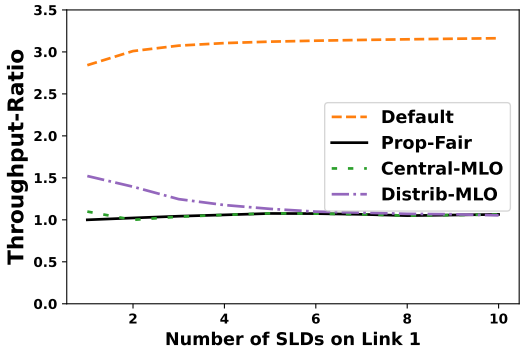
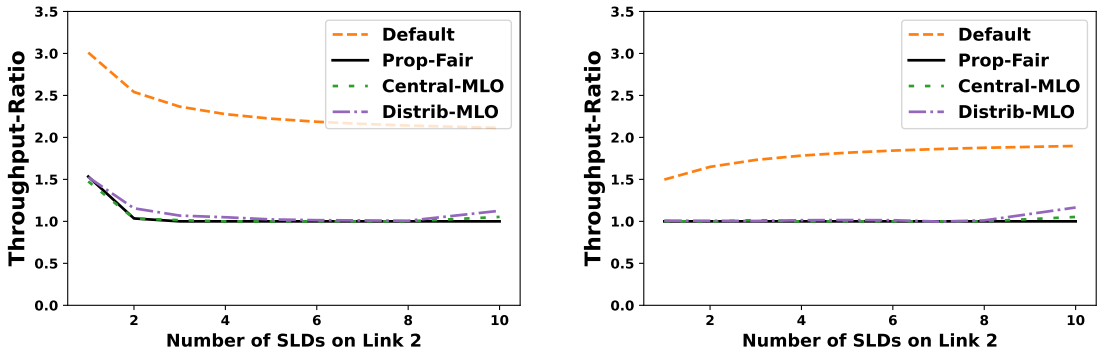
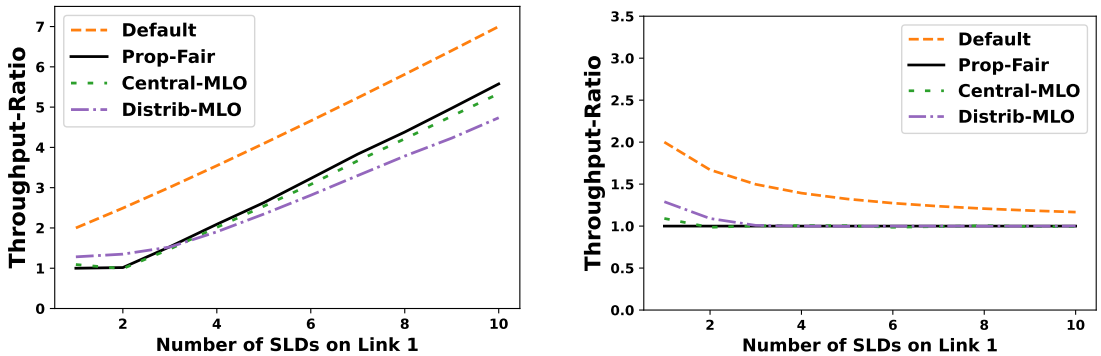


Figure 6.3: Simulation Configuration: nSld1 = nMld, nSld2 = 0



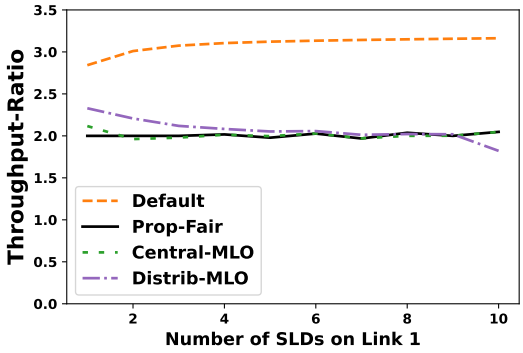
(a) Link 1 (b) Link 2

Figure 6.4: Simulation Configuration: nMld = nSld2 = nSld1 - 2



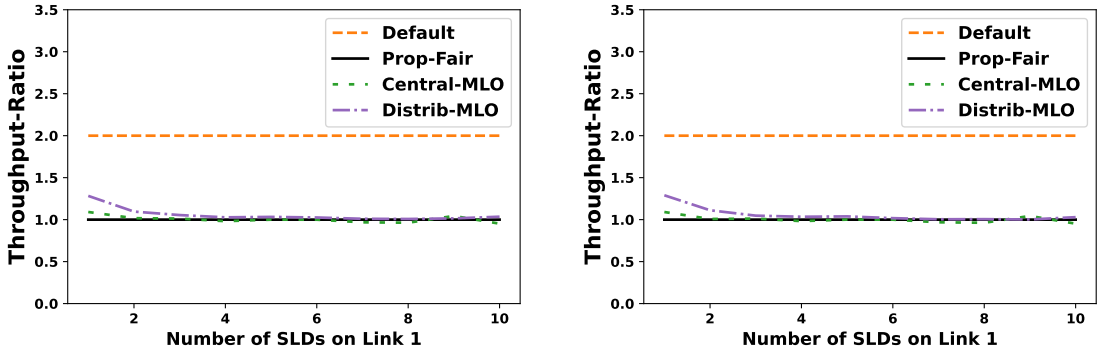
(a) Link 1 (b) Link 2

Figure 6.5: Simulation Configuration: nMld = nSld2 = 1



(a) Link 1

Figure 6.6: Simulation Configuration: $\gamma = 2$, nSld1 = nMld, nSld2 = 0



(a) Link 1 (b) Link 2

Figure 6.7: Simulation Configuration: nSld1 = nSld2 = nMld

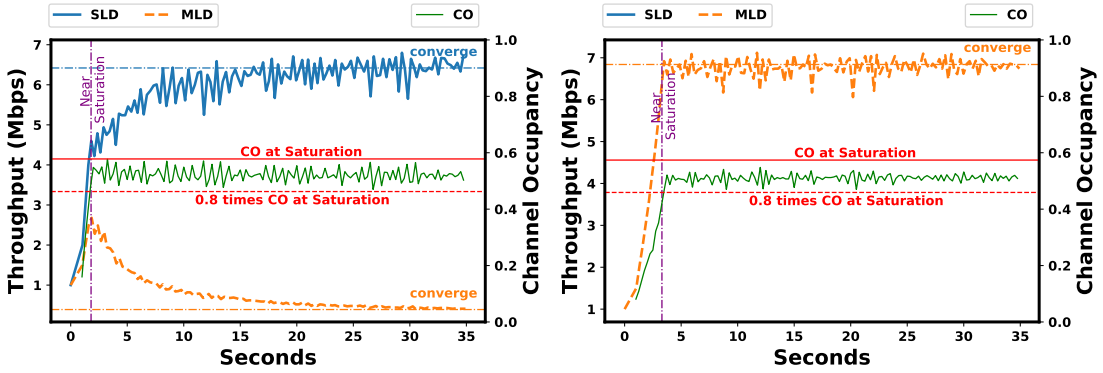
6.5.3 Analysis

Decentralized MLO approach is close to proportional fair throughput-ratio in majority of the simulations, however Decentralized MLO stays higher from the proportional fair ratio when there are less than 4 SLDs on a link. We observed that Decentralized MLO does not penalize MLDs enough when there are a few STAs on a link because the link's channel occupancy remains below saturation for substantial amount of time. With the increase in STAs, the channel occupancy reaches saturation more frequently, and consequently, MLDs are penalized sufficiently and a fair throughput ratio is achieved.

Hence, Decentralized MLO provides throughput-ratio sufficiently close to proportional fair value except when when there are only a few nodes on a link.

6.5.4 Convergence of Throughput Allocation

In this section, we evaluate the convergence of algorithm 2 through simulation results. Note that the algorithm is derived from equation (6.5) which has an asymptotic stable equilibrium point, i.e. the throughput should stop changing and converge to a value eventually with time.



(a) Link 1 (b) Link 2

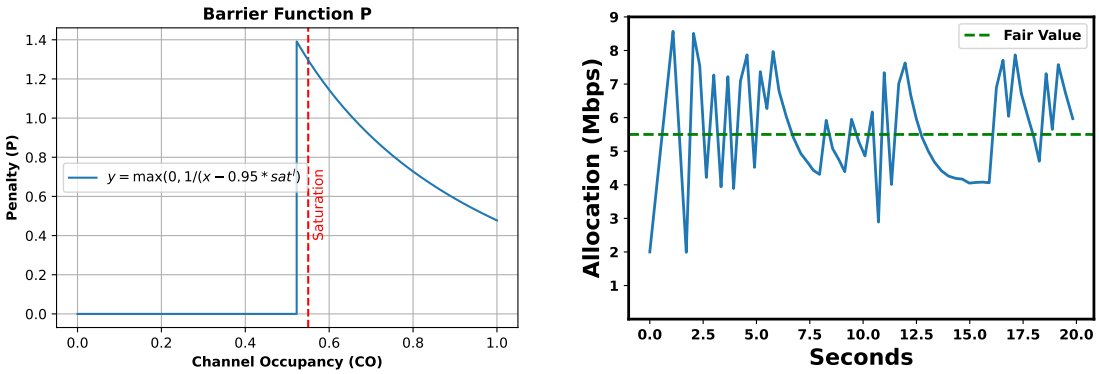
Figure 6.8: Convergence of throughput allocated to a SLD and a MLD on link 1 and 2 by Decentralized MLO in *ns-3* simulation.

Fig. 6.8 shows an example of how the algorithm converges to proportional fair throughput allocation for a SLD and a MLD. We have plotted trace of throughput and channel occupancy obtained from a simulation scenario with five SLDs on link 1, no SLD on link 2, and five MLDs in figure 6.3. We show graphs for one of the SLDs and one of the MLDs on two links to illustrate the convergence of the decentralized algorithm. In the figure, each node increases its throughput initially until channel occupancy approaches saturation. The rate of increase in throughput for MLD on link 1 is slower than SLD because MLD's combined throughput on both links, i.e. Th_i^{ϕ} in Eq (6.5), is higher than SLD. When channel occupancy is near saturation, we observe a see-saw pattern in throughput because nodes keep on making small adjustments to their throughput due to variations in channel occupancy feedback. Eventually, throughput converges near the optimal fair value. For all figures in 6.3-6.7, we measure the converged throughput by counting the number of packets received at the server in the last ten seconds of the simulation, assuming the throughput converges by the time we start counting in simulation.

We observed the impact of parameters 'S' and 'μ' on the convergence while tweaking

them in simulations. These two parameters require tweaking to converge throughput near the optimal fair value: a step size 'S' in equation (6.5) affects whether the algorithm will converge or not, and steepness of penalty 'μ' in equation (6.2) controls whether the converged value is fair or not. The rate of convergence is very slow if the step size is low, and on the other hand, if step size is high, then the see-saw pattern amplifies into oscillations, and the algorithm does not converge. Similarly, if μ is low then the penalty is not prohibitive enough and throughput overshoots fair value, and if it is high then the penalty is too prohibitive causing a noteworthy reduction in aggregate network throughput.

We also experimented with a barrier function P shown in figure 6.9a, but the throughput didn't converge as shown for an example in figure 6.9b.

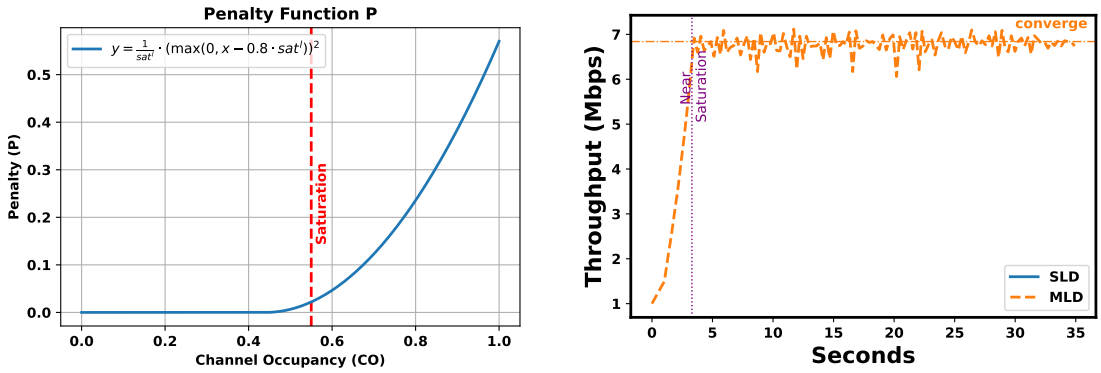


(a) Barrier Function

(b) Throughput

Figure 6.9: Throughput does not converge with a barrier function $P(co^l) = max(0, 1/(co^l - 0.95sat^l)$

We observed that $\Delta Th_i^{\Phi,l}$ were too high with the barrier function and it prevented throughput from converging. Hence, we replaced the barrier function with a smooth but steep enough penalty function in equation (6.2) and plotted in figure 6.10.



(a) Penalty Function

(b) Throughput

Figure 6.10: An example of a convering throughput when we replaced the barrier function with a smooth penalty function from equation (6.2)

6.6 Conclusion

In this chapter, we proposed and evaluated Decentralized MLO approach through simulations in *ns-3*, and found that it performs well except for scenarios where there are a few nodes on a link. We also studied some parameters, such as step size 'S', steepness 'μ', choice of penalty function 'P', that affect convergence of the decentralized algorithm 2.

Chapter 7

Channel Occupancy Helper In *ns-3*

7.1 Introduction

In the previous chapter 6, we defined Channel Occupancy metric 6.3 and used it in *ns-3* simulations 6.5.1. In this chapter, we describe how we implemented a new helper class in *ns-3* named as WifiCoTraceHelper to measure Channel Occupancy in *ns-3*. We show how we validated our implementation by comparing its results with an analytical model of DCF [4]. Then, we illustrate the broader usefulness of WifiCoTraceHelper in design and simulation of intelligent congestion-aware MLO schedulers through a few illustrative examples.

7.2 WifiCoTraceHelper Implementation in *ns-3*

Channel Occupancy is the fraction of time a link is sensed busy by a device over a time-window; within ns-3, it is measured as fraction for which the channel state is not sensed idle at PHY layer of a WiFi device, i.e.

$$\text{CO} = \frac{\text{Time-Window Duration} \ - \ \text{Idle Duration}}{\text{Time-Window Duration}} \tag{7.1}$$

Fig. 7.1 shows the overall working of our implementation of WifiCoTraceHelper, a helper class that measures Channel Occupancy, in *ns-3*. WifiPhy is an existing abstract class in *ns-3*, representing physical layer functions of WiFi devices and has a state machine that transitions between the states listed in Table 7.1 during a simulation [18]. On each state transition, WifiPhyStateHelper notifies the duration of time the WifiPhy resided in a state to listeners as shown in Fig. 7.1. We implemented WifiCoTraceHelper class that can *measure all the state durations* such as TX, RX, CCA_BUSY on all links of a WiFi device in ns-3.

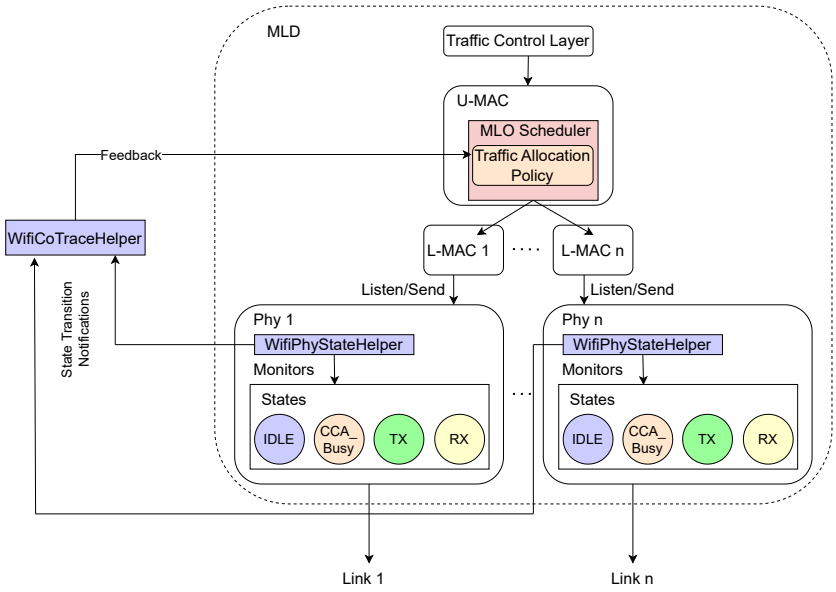


Figure 7.1: Interaction of WifiCoTraceHelper with other components in ns-3. The blue colored blocks highlight the components that enable the measurement of CO in ns-3.

WifiCoTraceHelper listens to these notifications and accumulates state duration in a map. It computes state durations from a node's viewpoint. While one node transmits data, the others receive it. Hence, the helper will receive TX notifications from the transmitting node but RX from the receiving nodes for same time interval. A single WifiCoTraceHelper instance can receive notifications from multiple nodes, and it will segregate them by nodes and links. The implementation is publicly accessible here[19] [1].

Table 7.1: WifiPhy States

States
IDLE
CCA_BUSY
RX
TX
SLEEP
SWITCHING
OFF

To measure state durations, we create an instance of WifiCoTraceHelper and pass a device as an argument to its `Enable` method. During a simulation, the helper instance accumulates the state durations which can be obtained from `GetDeviceRecords` method. The state durations are returned as a map of states-to-durations in a `DeviceRecord` class for each device. To calculate CO using the map of state durations, we sum all state durations in the map to calculate reference duration, get duration of IDLE state from the

[1]This implementation is available in *ns-3* from v3.44 onward.

map and apply the formula in Eq. 7.1.

The helper has three other useful features: `Start`, `Stop` and `Reset` methods. `Start` and `Stop` methods allow us to specify a time interval in simulation during which state duration should be measured[2]. `Reset` method discards all durations recorded till the instant it is called, and the helper will start recording durations afresh after it. This is useful in scenarios where a MLO scheduler requires link congestion in an immediate time-window instead of from the beginning of simulation; the MLO scheduler can thus get statistics and use it to reset the helper at periodic time instants.

7.2.1 Example Code Snippets

Using code snippets from an example program in [9], we demonstrate computation of Channel Occupancy using WifiCoTraceHelper in ns-3. We simulate a WiFi network with two nodes, an access point (AP) and a station (STA) with an uplink flow. We create an instance of WifiCoTraceHelper to measure state durations between 1 to 5 seconds during the simulation run:

```
Time start {Seconds(1.0)};
Time stop {Seconds(5.0)};
WifiCoTraceHelper wifiCoTraceHelper(start, stop);
```

WifiCoTraceHelper measures state durations for the entire simulation period if it is instantiated using the default constructor instead. Next, we enable tracing of state durations on both nodes using Enable method:

```
wifiCoTraceHelper.Enable(apNode);
wifiCoTraceHelper.Enable(staNode);
```

WifiCoTraceHelper accumulates state durations during the simulation run and we print the accumulated statistics for all nodes and links after the simulation is over using PrintStatistics method. Following is a snippet of the printed output for AP and STA on a link:

```
---- COT for AP:0#Link0 ---
Showing duration by states:
IDLE:       +2.54s   (63.42%)
```

[2]By default, the helper class keeps measuring the state duration data from start till the simulation ends.

```
CCA_BUSY: +74.02ms  (1.85%)
TX:       +83.06ms  (2.08%)
RX:        +1.31s   (32.65%)

---- COT for STA0:0#Link0 ---
Showing duration by states:
IDLE:      +2.54s   (63.42%)
CCA_BUSY: +25.38ms  (0.63%)
TX:        +1.39s   (34.66%)
RX:       +51.34ms  (1.28%)
```

Note in the snippet above that even though we measured state durations on both nodes for the same time interval, their statistics are not identical. STA spent the majority of time (34.66%) in TX, while AP spent the majority (32.66%) in RX, which is reasonable as the application flow is in uplink direction. In addition, both nodes stayed in idle state for same duration (63.42%).

Next we compute channel occupancy on a link as defined in equation (7.1). We obtain state durations of all nodes and links using GetDeviceRecords method and filter the state duration for AP on link 0 as follows:

```
auto& records=wifiCoTraceHelper.GetDeviceRecords();
size_t apNodeId = 0;
auto apDeviceRecord = std::find_if(records.begin(),records.end(),[
    apNodeId](auto& x) {
     return x.m_nodeId == apNodeId;
});
size_t link = 0;
auto linkDurations = apDeviceRecord->m_linkStateDurations.find(link);
```

We get idle state's duration, sum all state durations to calculate total duration, and apply equation (7.1) as follows:

```
auto& stats = linkDurations->second;
double idle = 0;
double total = 0;
for (auto& entry : stats)
{
    if (entry.first == WifiPhyState::IDLE)
    {
        idle = entry.second.GetDouble();
```

```
    }
    total += entry.second.GetDouble();
}
double channelOccupancy = (total - idle) / total;
```

Listing 7.1: Measure Channel Occupancy

The program terminates after printing channel occupancy's value as 0.3658 for this simulation scenario.

7.3 Analytical Validation

We used Markov chain model proposed by Bianchi in [4] to model the probabilities of channel access, successful transmission, etc; we use version in [15] that limits the number retries to 1 after reaching the maximum number of backoff stages.

As the Channel Occupancy will depend on the durations for which the channel is busy, it can be computed as:

$$co = \frac{P_s * (T^P_{data} + T_{data} + T^P_{Ack} + T_{Ack}) + P_c * (T^P_{data} + T_{data})}{P_e * T_e + P_s * T_s + P_c * T_c} \tag{7.2}$$

Table 7.2: List of Symbols.

Symbol	Description
T^P_{data}	PHY preamble duration of data packet
T^P_{Ack}	PHY preamble duration of ack packet
T_{data}	Transmission duration of data packet
T_{Ack}	Transmission duration of acknowledgement packet
T_e	Duration of empty slot = $9\mu s$
T_s	Duration of successful Tx
T_c	Duration of unsuccessful Tx due to collision
P_e	Probability of empty slot
P_s	Probability of successful Tx
P_c	Probability of collision

Eq. 7.2 takes into account the fraction of time for which the channel was busy. $P_s * (T^P_{data} + T_{data} + T^P_{Ack} + T_{Ack})$ will be the time for which the channel was busy due to the successful data transmission. It will involve the durations for PHY Preamble, the data which is the payload of the PHY layer involving MAC headers, and the Ack along

with its PHY preamble. $P_c * (T^P_{data} + T_{data})$ will be the time for which the channel was busy due to unsuccessful data transmission. Due to the unsuccessful attempt, this will involve only the transmission duration of the PHY Preamble and the data, not the Ack. The denominator represents the average duration over the different events involving idle time ($P_e * T_e$), successful transmission time ($P_s * T_s$) and collision time ($P_c * T_c$).

Next, we compare the CO measurements obtained from ns-3 simulation with the ones obtained from the above analytical model.

7.3.1 CO Measurements

Here we show the channel occupancy values obtained under saturated traffic conditions. Parameters for both analytical model and simulation are shown in Table 7.3.

Table 7.3: Simulation parameters

Parameter	**Value**
Payload Size	1000 Bytes
Channel Width	40MHz
Guard Interval	800 ns
MSDU Aggregation	Disabled
MPDU Aggregation	Disabled

Fig. 7.2 shows the channel occupancy obtained analytically and from ns-3 simulations for different numbers of users for MCS 3 and 11. The channel occupancy measurement from simulations very well matches with analytical models (with $< 4\%$ deviation). Note that even though the load is saturated, the channel occupancy is only about 0.5 for MCS 11. As aggregation is not considered, the overhead of idle and backoff states kick in leading to just ~ 0.5 channel occupancy.

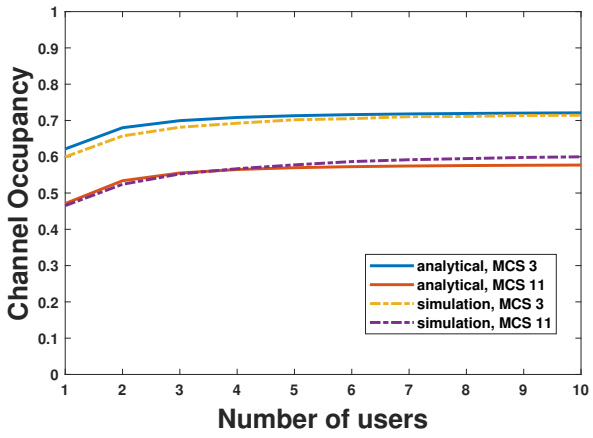


Figure 7.2: Channel Occupancy obtained analytically for different numbers of users with MCS 3 and 11.

The channel occupancy increases slightly with the increase in the number of users. Note that the CO increases with a decrease in MCS as the transmissions take longer on lower MCS.

One limitation of our implementation of CO is that it assumes that channel state is either IDLE or BUSY. In principle, nodes may also enter OFF, SLEEPING, SWITCHING states; however, we empirically observed that under saturated traffic, neither the access point nor the STAs went onto these states[3].

7.4 Other Channel Occupancy-Aware MLO Schedulers

With MLO, the MAC scheduler must decide how to allocate (application-layer) packets to the links. Prior works have mostly assumed *Greedy Scheduler* as the default, whereby any HOL (Head-of-Line) packet will be transmitted on whichever link first wins the contention. In general, there are multiple other MLO scheduling schemes such as SLCI, MLSA, MCAA [13], MCAB [12], LFTA [8], MH-RSAC [20] that represent the traffic allocation policy within the U-MAC layer. Notably, more advanced traffic allocation policies make use of link-state feedback (i.e. current and recent link-specific information) as input to the scheduling decisions. A common feedback parameter in [13, 12, 8, 20] is ***Channel Occupancy*** of the different links.

The design of an efficient and intelligent MLO scheduler is fundamental to achieving the benefits of MLO on WiFi7 networks. As such, given that MLD enabled devices are only recently entering deployments, there exists the need for a significantly deeper exploration of this topic via network performance evaluation. The popular open-source simulation platform ns-3 (www.nsnam.org) provides a credible WiFi stack, inclusive of MLO. While it presently lacks support for custom-built MLO schedulers, researchers have been using workarounds in support of their objectives. For example, [21] used tid-to-link mapping feature in WiFi7 to implement (random) traffic splitting scheduler, the same approach we described in 5.5.1. In this section, we showcase code snippets for two examples of how in-simulation measurement of metrics such as Channel Occupancy is useful in simulating and evaluating other MLO scheduler designs.

[3]Nodes likely go into these states when it utilizes power save mode.

7.4.1 Channel Occupancy-Aware Static MLO Scheduler

We show how to design an intelligent MLO scheduler that allocates traffic to links based on congestion. Such an MLO scheduler will schedule traffic on a link that is less occupied than other links. Prior work [13] proposes an MLO scheduler, namely SLCI (Single Link Less Congested Interface), that measures channel occupancy in a time window just before a flow arrival and schedules the flow on the least occupied link for its entire lifetime.

We demonstrate an application of WifiCoTraceHelper by simulating SLCI scheduling algorithm in a network with four uplink multi-link nodes (MLDs) and two links each. In our simulation, four uplink flows start on the network, one on each node. SLCI scheduling algorithm measures channel occupancies for a time-window of 0.5 seconds before each flow starts and schedules the flow on the least occupied link through this code snippet in the example program:

```
1 /* Start uplink flows at nodes every 2 seconds */
2 ...
3 WifiCoTraceHelper m_wificohelper;
4 m_wificohelper.Enable(sta);
5
6 for (size_t i = 1; i <= nWifi; i++)
7 {
8     Simulator::Schedule(Seconds(i * 2.0 - 0.5),
9                         &WifiCoTraceHelper::Reset,
10                        std::ref(m_wificohelper));
11    Simulator::Schedule(Seconds(i * 2.0),
12                        &StartFlow,
13                        std::ref(m_wificohelper),
14 ...
```

Listing 7.2: Reset Method

As shown in line 8 of the code snippet 7.2 above, we call Reset method 0.5 seconds before starting a flow so that we measure channel occupancy in an immediate time-window preceding flow arrival. We measure channel occupancies as shown in the code snippet 7.1 on both links of a node and compare them to select the least congested link as follows:

```
auto nodeId = sta->GetId();
```

```
auto co_link0 = ComputeChannelOccupancy(coHelper, nodeId, 0 /* linkId
    */);
auto co_link1 = ComputeChannelOccupancy(coHelper, nodeId, 1 /* linkId
    */);
auto selectedLinkId = (co_link0 < co_link1 ? 0 : 1);
```

Listing 7.3: SLCI's Link Selection

SLCI schedules an entire flow on the selected link and we simulate it in ns-3 using random splitting algorithm[21]. In this algorithm, we configure tid-to-link mapping during MLO setup such that tid '0' packets are sent over link '0' and tid '3' packets over link '1'.

```
std::string mapping = "0 0;1,2,3,4,5,6,7 1";
wifiDevice->GetMac()->GetEhtConfiguration()->SetAttribute("
    TidToLinkMappingNegSupport", EnumValue(
    WifiTidToLinkMappingNegSupport::ANY_LINK_SET));
wifiDevice->GetMac()->GetEhtConfiguration()->SetAttribute("
    TidToLinkMappingUl", StringValue(mapping));
wifiDevice->GetMac()->GetEhtConfiguration()->SetAttribute("
    TidToLinkMappingDl", StringValue(mapping));
```

Listing 7.4: MLO Tid-to-Link Mapping

Then we generate a flow by configuring PacketSocketClient application such that it generates tid '0' and tid '3' packets in a proportion called 'split-ratio' [4]. Tid '0' packets will be sent on link 0 and tid '3' on link 1 according to our tid-to-link mapping shown in code snippet 7.4. For SLCI, the split-ratio is either 1:0 or 0:1 because all packets of a flow are sent on either link '0' or link '1' respectively as follows:

```
/* Find probability of generating tid-3 packets to achieve split-
    ratio of either 1:0  or 0:1 */
double p_tid3 = (link == 0 ? 0.0 : 1.0);
auto client = CreateObject<PacketSocketClient>();
/* Tid 0 & 3 are AC_BE. */
auto tid0 = UintegerValue(0),
tid3 = UintegerValue(3);
client->SetAttribute("Priority", tid0);
client->SetAttribute("OptionalTid", tid3);
client->SetAttribute("OptionalTidPr",
```

[4]The feature to generate packets from two different tid is taken from [21]

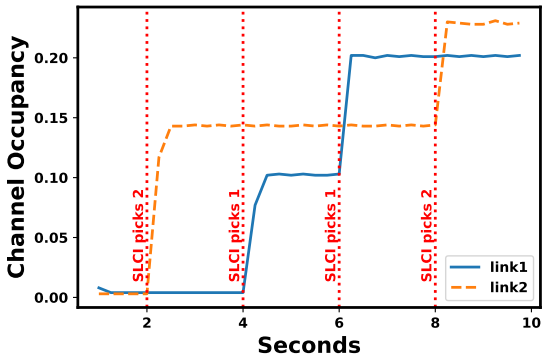


Figure 7.3: SLCI - Variation in channel occupancies of both links as new flows start at 2, 4, 6 and 8 seconds.

```
DoubleValue(p_tid3));
```

Listing 7.5: SLCI Split-Ratio

We simulated SLCI in ns-3 and Figure 7.3 shows variation in channel occupancies on the links during simulation. We can observe in the figure that channel occupancy on a link increases in steps as new flows are scheduled on it. SLCI schedules a new flow on a less occupied link as evident at time instants 4, 6 and 8 seconds in the figure. It chose link 1 at time-instants 4 and 6 seconds and link 2 at 8 seconds. The open-source program is publicly available at this Gitlab repository [9].

7.4.2 Channel Occupancy-Aware Dynamic MLO Scheduler

In the previous example, the allocation was static, i.e., once a flow is allocated to a link, it stays there for the lifetime of the flow. Further, the split of traffic was binary. We now show an example of how we can utilize WifiCoTraceHelper to obtain channel occupancy in a periodic fashion and update the scheduling decision dynamically. In a prior work, Multi-Link Congestion-aware Load balancing (MCAB) [12] is a dynamic MLO scheduling algorithm that, unlike SLCI, readjusts a non-binary split-ratio periodically. For simulating MCAB, we compute the split-ratio from channel occupancies as follows:

```
void MCAB_Scheduler(...) {
...
auto nodeId = sta->GetId();
auto co_link0 = ComputeChannelOccupancy(coHelper, nodeId, 0 /* LinkId
    */);
auto co_link1 = ComputeChannelOccupancy(coHelper, nodeId, 1 /* LinkId
    */);
```

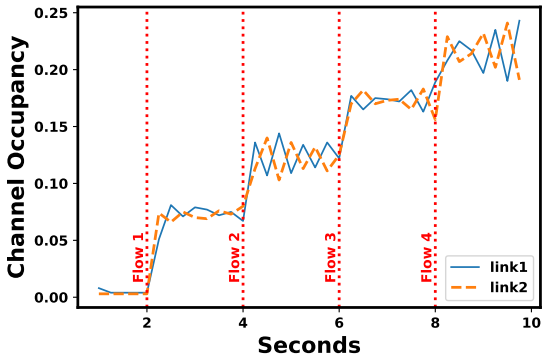


Figure 7.4: MCAB - Variation in channel occupancies of both links as new flows start at 2, 4, 6 and 8 seconds.

```
/* split-ratio = co_link1:co_link0 */
double p_tid3 = co_link0/(co_link0 + co_link1);
...
```

Listing 7.6: MCAB Split-Ratio

Like the previous simulation scenario for SLCI, we create four uplink multi-link nodes (MLDs) and two links each. Four uplink flows start on the network, one on each node. In our simulation example [9], MCAB measures channel occupancies in the latest time-window and adjusts split-ratio periodically every 0.25 seconds as follows:

```
void MCAB_Scheduler(WifiCoTraceHelper& coHelper, Ptr<Node> sta, Ptr<
    PacketSocketClient> client)
{
...
client->SetAttribute("OptionalTidPr",
                     DoubleValue(p_tid3));
coHelper.Reset();
Time periodicity{Seconds(0.25)};
Simulator::Schedule(periodicity,
&MCAB_Scheduler, std::ref(coHelper), sta, client);
}
```

Listing 7.7: MCAB's Periodicity

We replace SLCI with MCAB in the previous simulation example and compare their channel occupancy curves in Fig. 7.3 and Fig. 7.4. The two channel occupancy curves in Fig. 7.4 are close to each other for the entire simulation which shows that MCAB splits traffic almost evenly on the two links. In comparison, SLCI has wide gaps between the two curves in Fig. 7.3. MCAB has a seesaw pattern in curves while SLCI has smooth

horizontal curves because MCAB readjusts split-ratio every 0.25 seconds, but SLCI does not.

7.5 Conclusion

We implemented `WifiCoTraceHelper` class to measure Channel Occupancy in *ns-3* and validated it against Bianchi's analytical model of DCF [4]. We used `WifiCoTraceHelper` not only in our simulations of Decentralized MLO approach in chapter 6, but also showcased its broader applicability by simulating other MLO scheduler designs proposed in prior works [13, 12].

Chapter 8

Conclusion

In this work, both Centralized MLO and Decentralized MLO approaches are evaluated through simulations in simple, symmetrical scenarios. So we conclude with suggesting some possible enhancements to this work required for application in a real-world scenario.

8.1 Future Work

In Decentralized MLO approach, we found parameters, such as steepness μ in the penalty function, through manual trial-and-error in simulations. In extension of this work, we can come up with an automated way to find the steepness parameter for any simulation scenario.

In this work, the simulation parameters on all nodes are set to same values to simplify analysis and to focus on the inherent difference between multi-link and single-link operations. In future, we can simulate on more realistic scenarios such as nodes with different MCS, packet aggregation etc. We anticipate some challenges in extending current approaches to such simulation scenarios. One challenge could be determination of saturated throughput or saturated channel occupancy (sat^l) on a link with heterogeneous nodes. To mitigate this challenge, we suggest two approaches. One approach is to determine the maximum achievable throughput in a heterogeneous network through some control algorithm on AP on-the-fly or compute a rough estimate beforehand. Second approach could be fair allocation of airtime (or channel occupancy) instead of throughput. The benefit of modeling distribution of channel occupancy instead of throughput is that we know the maximum channel occupancy is 1.0 ideally or 0.9 pragmatically, so we do not need any analytical model to compute the saturated channel occupancy in a realistic scenario.

Approach	Advantage	Disadvantage
Centralized MLO	It is very close to proportional fairness in all simulation scenarios. It is simple and should be easy to implement in practice because it does not have any parameters to tweak.	AP should assume extra responsibility to control throughput allocation of the network.
Decentralized MLO	It is close to proportional fairness in majority of simulation scenarios. Each node can determine its throughput allocation by sniffing Channel Occupancy and applying gradient ascent algorithm, so AP does not assume extra responsibility of throughput allocation.	Throughput ratio is higher than proportional fair value by a small amount when number of nodes on a link are less than four. It is harder to implement due to requirement of fine-tuning the penalty function.

Table 8.1: Comparison of Centralized MLO and Decentralized MLO approaches

8.2 Summary

The simulation results show that compared to the default benchmark, the case when the simulation is run with greedy MLO scheduler, both Centralized MLO and Decentralized MLO approaches improve fairness and are close to optimal proportional fairness obtained via NUM problem formulation. Centralized MLO approach, in particular, is always close to the optimal fairness ratio with a deviation of 9.2% in the worst case. Decentralized MLO approach is very close to the proportional fairness in almost all scenarios, except when there are only a few SLDs and MLDs on a link. This deviation in Decentralized MLO approach is because MLDs converge to a higher throughput than its fair share when it is competing with only a few nodes for channel occupancy on a link. In summary, both approaches improve fairness substantially compared to the default benchmark and Centralized MLO comes closest to proportional fairness.

We find Centralized MLO approach to be the easiest to implement in simulation and we believe that it should be the easiest to implement in practice as well. Though Decentralized MLO has the advantage that it does not require central coordination, we discovered its unique challenges of tweaking parameters in the penalty function. The fine-tuning of parameters to obtain good results will only get harder in practice compared to simulations. In summary, it is efficient to solve the convex optimization problem for Centralized MLO, and Centralized MLO is not complicated by the necessity to tweak parameters like in Decentralized MLO.

The relative advantages and disadvantages of Centralized MLO and Decentralized MLO approaches are summarized in table 8.1.

References

[1] *Multi-link Architecture*. 2024. URL: https://slideplayer.com/slide/17974909/ (visited on 10/30/2024).

[2] Shubhodeep Adhikari and Sindhu Verma. "Analysis of Multilink in IEEE 802.11 be". In: *IEEE Communications Standards Magazine* 6.3 (2022), pp. 52–58.

[3] Mohammed Amer, Anthony Busson, and Isabelle Guérin Lassous. "Association optimization based on access fairness for Wi-Fi networks". In: *Computer Networks* 137 (2018), pp. 173–188. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2018.03.004. URL: https://www.sciencedirect.com/science/article/pii/S1389128618301129.

[4] Giuseppe Bianchi. "Performance analysis of the IEEE 802.11 distributed coordination function". In: *IEEE Journal on selected areas in communications* 18.3 (2000), pp. 535–547.

[5] Marc Carrascosa-Zamacois et al. "Wi-Fi Multi-Link Operation: An Experimental Study of Latency and Throughput". In: *IEEE/ACM Transactions on Networking* (2023), pp. 1–15.

[6] *CVXPY: Convex solver library in python*. https://www.cvxpy.org/. [Online; accessed 11-May-2025].

[7] Yayu Gao and Sumit Roy. "Achieving proportional fairness for LTE-LAA and Wi-Fi coexistence in unlicensed spectrum". In: *IEEE Transactions on Wireless Communications* 19.5 (2020), pp. 3390–3404.

[8] Jagrati Kulshrestha et al. "LFTA: Legacy Friendly Traffic Allocation Strategy for Multi-Link Operation in WiFi7". In: *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. 2024, pp. 448–456. DOI: 10.1109/COMSNETS59351.2024.10426831.

[9] Puneet Kumar. *Example Programs of WifiCoTraceHelper*. https://gitlab.com/puneet23067/ns-3-dev/-/tree/wifi-co-examples/scratch?ref_type=heads. [Online; accessed 11-Nov-2024]. 2024.

[10] Puneet Kumar. *Simulation Programs in ns-3*. https://gitlab.com/puneet23067/ns-3-dev/-/tree/wifi-pf-simulations/scratch?ref_type=heads. [Online; accessed 29-Nov-2024]. 2024.

[11] Yuan Le et al. "A time fairness-based MAC algorithm for throughput maximization in 802.11 networks". In: *IEEE Transactions on Computers* 64.1 (2013), pp. 19–31.

[12] Álvaro López-Raventós and Boris Bellalta. "Dynamic Traffic Allocation in IEEE 802.11 be Multi-link WLANs". In: *IEEE Wireless Communications Letters* (2022).

[13] Álvaro López-Raventós and Boris Bellalta. “IEEE 802.11 be multi-link operation: When the best could be to use only a single interface”. In: *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*. IEEE. 2021, pp. 1–7.

[14] Álvaro López-Raventós and Boris Bellalta. “Multi-link operation in IEEE 802.11 be WLANs”. In: *IEEE Wireless Communications* 29.4 (2022), pp. 94–100.

[15] Morteza Mehrnoush et al. “Analytical modeling of Wi-Fi and LTE-LAA coexistence: Throughput and impact of energy detection threshold”. In: *IEEE/ACM Transactions on Networking* 26.4 (2018), pp. 1990–2003.

[16] Morteza Mehrnoush et al. “On the fairness of Wi-Fi and LTE-LAA coexistence”. In: *IEEE Transactions on Cognitive Communications and Networking* 4.4 (2018), pp. 735–748.

[17] Gaurang Naik, Dennis Ogbe, and Jung-Min Jerry Park. “Can Wi-Fi 7 Support Real-Time Applications? On the Impact of Multi Link Aggregation on Latency”. In: *ICC 2021 - IEEE International Conference on Communications*. 2021, pp. 1–6. DOI: 10.1109/ICC42927.2021.9500256.

[18] *WifiCoTraceHelper ns-3 User Documentation*. https://www.nsnam.org/docs/models/html/wifi-user.html. [Online; accessed 16-May-2025].

[19] *WifiCoTraceHelper ns-3 Class Reference*. https://www.nsnam.org/docs/release/3.44/doxygen/d1/d95/classns3_1_1_wifi_co_trace_helper.html. [Online; accessed 11-May-2025].

[20] Pedro Enrique Iturria Rivera et al. “RL meets Multi-Link Operation in IEEE 802.11 be: Multi-Headed Recurrent Soft-Actor Critic-based Traffic Allocation”. In: *arXiv preprint arXiv:2303.08959* (2023).

[21] Muyuan Shen et al. “Delay in Multi-Link Operation in ns-3: Validation and Impact of Traffic Splitting”. en. In: *2024 Workshop on ns-3*. Barcelona Spain: ACM, June 2024, pp. 19–26. ISBN: 9798400717635. DOI: 10.1145/3659111.3659116. URL: https://dl.acm.org/doi/10.1145/3659111.3659116 (visited on 10/16/2024).

[22] R. Srikant. *The Mathematics of Internet Congestion Control*. Systems & Control: Foundations & Applications. Boston, MA: Birkhäuser Boston, 2004. ISBN: 978-1-4612-6498-9 978-0-8176-8216-3. DOI: 10.1007/978-0-8176-8216-3. URL: http://link.springer.com/10.1007/978-0-8176-8216-3 (visited on 10/18/2024).

[23] R. Srikant and Lei Ying. *Communication Networks: An Optimization, Control, and Stochastic Networks Perspective*. Cambridge University Press, 2013.

[24] Xinghua Sun and Lin Dai. “Towards fair and efficient spectrum sharing between LTE and WiFi in unlicensed bands: Fairness-constrained throughput maximization”. In: *IEEE Transactions on Wireless Communications* 19.4 (2020), pp. 2713–2727.

[25] *Wi-Fi 6, Wi-Fi 6E And Wi-Fi 7 Chipset Market Size, Share and Trends Analysis Report By Chipset Type (Wi-Fi 6, Wi-Fi 6E, W-Fi 7), By Device Type, By Application, By Region, And Segment Forecasts, 2023 - 2030*. URL: https://www.grandviewresearch.com/industry-analysis/wi-fi-6-wi-fi-6e-chipset-market-report (visited on 10/30/2024).

List of Publications

[1] Puneet Kumar et al. “Use of Channel Occupancy for Multi Link WiFi 7 Scheduler Design in ns-3”. In: *2025 17th International Conference on COMmunication Systems and NETworks (COMSNETS)*. ISSN: 2155-2509. Jan. 2025, pp. 1211–1215. DOI: 10.1109/COMSNETS63942.2025.10885613. URL: https://ieeexplore.ieee.org/document/10885613 (visited on 04/29/2025).

Bibliography

[1] Shubhodeep Adhikari and Sindhu Verma. “Analysis of Multilink in IEEE 802.11 be”. In: *IEEE Communications Standards Magazine* 6.3 (2022), pp. 52–58.

[2] Mohammed Amer, Anthony Busson, and Isabelle Guérin Lassous. “Association optimization based on access fairness for Wi-Fi networks”. In: *Computer Networks* 137 (2018), pp. 173–188. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2018.03.004. URL: https://www.sciencedirect.com/science/article/pii/S1389128618301129.

[3] Evgeny Avdotin et al. “Enabling massive real-time applications in IEEE 802.11 be networks”. In: *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE. 2019, pp. 1–6.

[4] Evgeny Avdotin et al. “Resource allocation strategies for real-time applications in Wi-Fi 7”. In: *2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. IEEE. 2020, pp. 1–6.

[5] Sergio Barrachina-Muñoz, Boris Bellalta, and Edward Knightly. “Wi-Fi all-channel analyzer”. In: *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*. 2020, pp. 72–79.

[6] Boris Bellalta et al. “Delay Analysis of IEEE 802.11 be multi-link operation under finite load”. In: *IEEE Wireless Communications Letters* 12.4 (2023), pp. 595–599.

[7] Giuseppe Bianchi. “Performance analysis of the IEEE 802.11 distributed coordination function”. In: *IEEE Journal on selected areas in communications* 18.3 (2000), pp. 535–547.

[8] Holger Boche, Marcin Wiczanowski, and Slawomir Stanczak. “Unifying view on min-max fairness, max-min fairness, and utility optimization in cellular networks”. In: *EURASIP Journal on wireless communications and networking* 2007 (2007), pp. 1–20.

[9] Cristina Cano et al. “Fair coexistence of scheduled and random access wireless networks: Unlicensed LTE/WiFi”. In: *IEEE/ACM Transactions on Networking* 25.6 (2017), pp. 3267–3281.

[10] Marc Carrascosa et al. “An experimental study of latency for IEEE 802.11 be multi-link operation”. In: *ICC 2022-IEEE International Conference on Communications*. IEEE. 2022, pp. 2507–2512.

[11] Marc Carrascosa et al. “An Experimental Study of Latency for IEEE 802.11be Multi-link Operation”. In: *ICC 2022 - IEEE International Conference on Communications*. ISSN: 1938-1883. May 2022, pp. 2507–2512. DOI: 10.1109/ICC45855.2022.9838765. URL: https://ieeexplore.ieee.org/document/9838765 (visited on 03/20/2024).

[12] Marc Carrascosa-Zamacois et al. “Understanding Multi-link Operation in Wi-Fi 7: Performance, Anomalies, and Solutions”. In: *arXiv preprint arXiv:2210.07695* (2022).

[13] Cailian Deng et al. “IEEE 802.11 be Wi-Fi 7: New challenges and opportunities”. In: *IEEE Communications Surveys and Tutorials* 22.4 (2020), pp. 2136–2166.

[14] *Fairness Measure*. [accessed 10-July-2020]. URL: https://en.wikipedia.org/wiki/Fairness_measure.

[15] Yayu Gao and Sumit Roy. “Achieving proportional fairness for LTE-LAA and Wi-Fi coexistence in unlicensed spectrum”. In: *IEEE Transactions on Wireless Communications* 19.5 (2020), pp. 3390–3404.

[16] *Global Economic Value of Wi-Fi® 2021 – 2025*. URL: https://www.wi-fi.org/system/files/Global_Economic_Value_of_Wi-Fi_2021-2025_202109.pdf (visited on 08/30/2024).

[17] Cengis Hasan, Mahesh K Marina, and Ursula Challita. “On LTE-WiFi coexistence and inter-operator spectrum sharing in unlicensed bands: Altruism, cooperation and fairness”. In: *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 2016, pp. 111–120.

[18] Hongli He et al. “Proportional fairness-based resource allocation for LTE-U coexisting with Wi-Fi”. In: *IEEE Access* 5 (2016), pp. 4720–4731.

[19] Roger Pierre Fabris Hoefel. “IEEE 802.11 be: throughput and reliability enhancements for next Generation Wi-Fi networks”. In: *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE. 2020, pp. 1–7.

[20] Haonan Hu et al. “On the fairness of the coexisting LTE-U and WiFi networks sharing multiple unlicensed channels”. In: *IEEE Transactions on Vehicular Technology* 69.11 (2020), pp. 13890–13904.

[21] Kaiwen Huang et al. “Mutli-link channel access schemes for IEEE 802.11 be extremely high throughput”. In: *IEEE Communications Standards Magazine* 6.3 (2022), pp. 46–51.

[22] Rajendra K Jain, Dah-Ming W Chiu, William R Hawe, et al. “A quantitative measure of fairness and discrimination”. In: *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA* 21 (1984).

[23] Li Bin Jiang and Soung Chang Liew. “Proportional fairness in wireless LANs and ad hoc networks”. In: *IEEE Wireless Communications and Networking Conference, 2005*. Vol. 3. 2005, 1551–1556 Vol. 3. DOI: 10.1109/WCNC.2005.1424745.

[24] Tarun Joshi et al. “Airtime fairness for IEEE 802.11 multirate networks”. In: *IEEE Transactions on Mobile Computing* 7.4 (2008), pp. 513–527.

[25] Aliasghar Keyhanian et al. “Analyzing the coexistence of Wi-Fi and LAA-LTE towards a proportional throughput fairness”. In: *Proceedings of the 16th ACM International Symposium on Mobility Management and Wireless Access*. 2018, pp. 95–101.

[26] Evgeny Khorov, Ilya Levitsky, and Ian F Akyildiz. “Current status and directions of IEEE 802.11 be, the future Wi-Fi 7”. In: *IEEE access* 8 (2020), pp. 88664–88688.

[27] Nikolay Korolev, Ilya Levitsky, and Evgeny Khorov. “Analyses of NSTR multi-link operation in the presence of legacy devices in an IEEE 802.11 be network”. In: *2021 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE. 2021, pp. 94–98.

[28] Nikolay Korolev, Ilya Levitsky, and Evgeny Khorov. “Analytical model of multi-link operation in saturated heterogeneous Wi-Fi 7 networks”. In: *IEEE Wireless Communications Letters* 11.12 (2022), pp. 2546–2549.

[29] Nikolay Korolev et al. “Study of multi-link channel access without simultaneous transmit and receive in IEEE 802.11 be networks”. In: *IEEE Access* 10 (2022), pp. 126339–126351.

[30] Jagrati Kulshrestha et al. “LFTA: Legacy Friendly Traffic Allocation Strategy for Multi-Link Operation in WiFi7”. In: *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. 2024, pp. 448–456. DOI: 10.1109/COMSNETS59351.2024.10426831.

[31] Massimiliano Laddomada et al. “On the throughput performance of multirate IEEE 802.11 networks with variable-loaded stations: analysis, modeling, and a novel proportional fairness criterion”. In: *IEEE Transactions on Wireless Communications* 9.5 (2010), pp. 1594–1607.

[32] Xiyang Lan, Xinyu Zu, Jie Yang, et al. “Enhanced Multilink Single-Radio Operation for the Next-Generation IEEE 802.11 BE Wi-Fi Systems”. In: *Security and Communication Networks* 2022 (2022), pp. 1–11.

[33] Yuan Le et al. “A time fairness-based MAC algorithm for throughput maximization in 802.11 networks”. In: *IEEE Transactions on Computers* 64.1 (2013), pp. 19–31.

[34] L. Li, M. Pal, and Y. R. Yang. “Proportional Fairness in Multi-Rate Wireless LANs”. In: *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*. 2008, pp. 1004–1012. DOI: 10.1109/INFOCOM.2008.154.

[35] David López-Pérez et al. “IEEE 802.11 be extremely high throughput: The next generation of Wi-Fi technology beyond 802.11 ax”. In: *IEEE Communications Magazine* 57.9 (2019), pp. 113–119.

[36] Álvaro López-Raventós and Boris Bellalta. “Dynamic Traffic Allocation in IEEE 802.11 be Multi-link WLANs”. In: *IEEE Wireless Communications Letters* (2022).

[37] Álvaro López-Raventós and Boris Bellalta. “IEEE 802.11 be multi-link operation: When the best could be to use only a single interface”. In: *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*. IEEE. 2021, pp. 1–7.

[38] Álvaro López-Raventós and Boris Bellalta. “Multi-link operation in IEEE 802.11 be WLANs”. In: *IEEE Wireless Communications* 29.4 (2022), pp. 94–100.

[39] Daniele Medda et al. “Investigating Inclusiveness and Backward Compatibility of IEEE 802.11 be Multi-link Operation”. In: *2022 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE. 2022, pp. 20–24.

[40] Morteza Mehrnoush et al. “Analytical modeling of Wi-Fi and LTE-LAA coexistence: Throughput and impact of energy detection threshold”. In: *IEEE/ACM Transactions on Networking* 26.4 (2018), pp. 1990–2003.

[41] Morteza Mehrnoush et al. “On the fairness of Wi-Fi and LTE-LAA coexistence”. In: *IEEE Transactions on Cognitive Communications and Networking* 4.4 (2018), pp. 735–748.

[42] *Multi-link Architecture*. URL: https://slideplayer.com/slide/17974909/ (visited on 10/30/2024).

[43] Wisnu Murti and Ji-Hoon Yun. “Multi-link operation with enhanced synchronous channel access in IEEE 802.11 be wireless LANs: Coexistence issue and solutions”. In: *Sensors* 21.23 (2021), p. 7974.

[44] Wisnu Murti and Ji-Hoon Yun. “Multilink operation in IEEE 802.11 be wireless LANs: Backoff overflow problem and solutions”. In: *Sensors* 22.9 (2022), p. 3501.

[45] Gaurang Naik, Dennis Ogbe, and Jung-Min Jerry Park. “Can Wi-Fi 7 support real-time applications? On the impact of multi link aggregation on latency”. In: *ICC 2021-IEEE International Conference on Communications*. IEEE. 2021, pp. 1–6.

[46] Sharan Naribole, Srinivas Kandala, and Ashok Ranganath. “Multi-channel mobile access point in next-generation IEEE 802.11 be WLANs”. In: *ICC 2021-IEEE International Conference on Communications*. IEEE. 2021, pp. 1–7.

[47] Hyunhee Park and Cheolwoo You. “Latency impact for massive real-time applications on multi link operation”. In: *2021 IEEE Region 10 Symposium (TENSYMP)*. IEEE. 2021, pp. 1–5.

[48] Oran Sharon and Yaron Alpert. “Scheduling strategies and throughput optimization for the Downlink for IEEE 802.11 ax and IEEE 802.11 ac based networks”. In: *arXiv preprint arXiv:1709.04818* (2017).

[49] Muyuan Shen et al. “Delay in Multi-Link Operation in ns-3: Validation and Impact of Traffic Splitting”. en. In: *2024 Workshop on ns-3*. Barcelona Spain: ACM, June 2024, pp. 19–26. ISBN: 9798400717635. DOI: 10.1145/3659111.3659116. URL: https://dl.acm.org/doi/10.1145/3659111.3659116 (visited on 10/16/2024).

[50] Huaizhou SHI et al. “Fairness in Wireless Networks:Issues, Measures and Challenges”. In: *IEEE Communications Surveys and Tutorials* 16.1 (2014), pp. 5–24. DOI: 10.1109/SURV.2013.050113.00015.

[51] Vasilios A Siris and George Stamatakis. “Optimal CWmin selection for achieving proportional fairness in multi-rate 802.11 e WLANs: test-bed implementation and evaluation”. In: *Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization*. 2006, pp. 41–48.

[52] Taewon Song and Taeyoon Kim. “Performance analysis of synchronous multi-radio multi-link MAC protocols in IEEE 802.11 be extremely high throughput WLANs”. In: *Applied Sciences* 11.1 (2020), p. 317.

[53] R. Srikant. *The Mathematics of Internet Congestion Control*. Systems & Control: Foundations & Applications. Boston, MA: Birkhäuser Boston, 2004. ISBN: 978-1-4612-6498-9 978-0-8176-8216-3. DOI: 10.1007/978-0-8176-8216-3. URL: http://link.springer.com/10.1007/978-0-8176-8216-3 (visited on 10/18/2024).

[54] R. Srikant and Lei Ying. *Communication Networks: An Optimization, Control, and Stochastic Networks Perspective*. Cambridge University Press, 2013.

[55] Xinghua Sun and Lin Dai. “Towards fair and efficient spectrum sharing between LTE and WiFi in unlicensed bands: Fairness-constrained throughput maximization”. In: *IEEE Transactions on Wireless Communications* 19.4 (2020), pp. 2713–2727.

[56] Ching-Lun Tai, Shyam Krishnan Venkateswaran, and Raghupathy Sivakumar. "Equitas: Fairness-Aware Dynamic Link Selection for EMLSR Operation in IEEE 802.11 be". In: *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*. 2024, pp. 1–6.

[57] Godfrey Tan and John V Guttag. "Time-based Fairness Improves Performance in Multi-Rate WLANs." In: *USENIX annual technical conference, general track*. 2004, pp. 269–282.

[58] Xijun Wang et al. "Throughput and fairness analysis of Wi-Fi and LTE-U in unlicensed band". In: *IEEE Journal on Selected Areas in Communications* 35.1 (2016), pp. 63–78.

[59] *What Is Wi-Fi 6E?* URL: https://www.cisco.com/c/en_in/products/wireless/what-is-wi-fi-6e.html (visited on 04/30/2023).

[60] *Wi-Fi 6, Wi-Fi 6E And Wi-Fi 7 Chipset Market Size, Share and Trends Analysis Report By Chipset Type (Wi-Fi 6, Wi-Fi 6E, W-Fi 7), By Device Type, By Application, By Region, And Segment Forecasts, 2023 - 2030*. URL: https://www.grandviewresearch.com/industry-analysis/wi-fi-6-wi-fi-6e-chipset-market-report (visited on 10/30/2024).

[61] *Wi-Fi 6E*. URL: https://www.intel.com/content/www/us/en/products/docs/wireless/wi-fi-6e.html (visited on 04/30/2023).

[62] *WiFi 7 and the controversy over 6 GHz unlicensed vs licensed spectrum*. URL: https://techblog.comsoc.org/2023/12/26/wifi-7-and-the-controversy-over-6-ghz-unlicensed-vs-licensed-spectrum/ (visited on 10/30/2024).

[63] Mao Yang and Bo Li. "Survey and perspective on extremely high throughput (EHT) WLAN—IEEE 802.11 be". In: *Mobile Networks and Applications* 25 (2020), pp. 1765–1780.

[64] Jie Zhang et al. "WiFi 7 with Different Multi-Link Channel Access Schemes: Modeling, Fairness and Optimization". In: *IEEE Transactions on Communications* (2024).

Curriculum Vitae

Puneet Kumar

Email: puneet23067@iiitd.ac.in | GitLab: https://gitlab.com/puneet23067/ns-3-dev

Education

2023–2025: M.Tech. in Computer Science, Indraprastha Institute of Information Technology, Delhi

• **CGPA:** 9.9 / 10.0

2007–2011: B.E. in Computer Engineering, Netaji Subhas Institute of Technology, Delhi

• **Percent:** 73.4. / 100.0

Experience

2023–2025: Network and Security Lab, IIIT Delhi

• **Focus:** Study a new feature called Multi-link operation (MLO) in WiFi 7 standard through simulations in *ns-3*.

2011–2023: Software Development Engineer, Amazon India

• **Work Experience:** Develop full-stack distributed web applications in Amazon's cloud computing platform called AWS. Work in multiple business domains such as payments, retail website, catalog enrichment and advertisement moderation. Design and implement a variety of applications like microservices, frontend, web crawler, big data management, data analytics.

Skills

Languages: Java, Python, C++

Tools: AWS, *ns-3*, Apache Spark, LLVM

Awards

2025: UGC NET 2025 qualified for PhD and Assistant Professor with a percentile of 99.98 in Computer Science and Applications.

2024: Received Dean's List Award for academic excellence for securing a SGPA above 9.5 in academic year 2023-24 at IIITD.