POLITECNICO DI TORINO

Master's degree in
Communications and Computer Networks Engineering

Master's Degree Thesis

Performance evaluation of Networked Music Performance (NMP) over 5G scenario through a Python-based simulator

Supervisor:
Prof.ssa Cristina Rottondi

Candidate:
Angelo Riccobene

a.y. 2023-24

Abstract

Networked Music Performances (NMPs) enable remote musicians to perform together by means of low-latency audio/video streaming over a telecommunication network. Latency and reliability are two key parameters for the end-to-end transmission of audio information through networks in this context. The stringent requirements of NMPs with respect to those parameters pose a major challenge for 5G networks, which are expected to deliver significant Key Performance Indicator (KPI) improvements with respect to 4G.

The objective of this thesis is to develop a Python-based simulator to define the placement of virtual machines operating as audio mixing servers within a set of candidate network nodes, while complying to the latency requirements of NMPs. The simulator emulates an optical network functioning as the backend for a 5G access network. Several users are connected to the 5G network, participating in remote musical interactions to simulate a Networked Music Performance scenario. This study aims to assess the performance and efficiency of the heuristic method, benchmarking it against optimal solutions derived from two Mixed Integer Linear Programming (MILP) models in terms of end-to-end latency, server load and computational timings.

Contents

List of Tables

List of Figures

Acronyms

C-RAN	Centralized Radio Access Network
CDF	Cumulative Distribution Function
CDN	Content Delivery Network
CN	Core Network
CU	Centralized Unit
DRAN	Distributed Radio Access Network
DU	Distributed Unit
eMBB	enhanced Mobile Broadband
gNB	Next-generation NodeB
IoMusT	Internet of Musical Things
IoT	Internet of Things
KPI	Key Performance Indicator
ML	Machine Learning
MEC	Multi-access Edge Computing
MILP	Mixed Integer Linear Programming
MILPs	Mixed Integer Linear Programs
MIMO	Multiple-Input Multiple-Output
mMTC	massive Machine-Type Communications
NMP	Networked Music Performance
NMPs	Networked Music Performances
NSA	Non-Standalone

O-RAN	Open RAN Alliance
ODN	Optical Distribution Network
OLT	Optical Line Terminal
ONTs	Optical Network Terminals
ONUs	Optical Network Units
PON	Passive Optical Network
PONs	Passive Optical Networks
PtP	Point-to-Point
QoS	Quality of Service
RAN	Radio Access Network
RU	Remote Unit
SA	Standalone
SDN	Software-Defined Networking
SMIs	Smart Musical Instruments
TDM-PON	Time Division Multiplexing-Passive Optical Networks
UEs	User Equipments
URLLC	Ultra-Reliable Low Latency Communications
VM	Virtual Machine
WAN	Wide-area Network
WDM	Wavelength Division Multiplexing

Chapter 1

Introduction

Nowadays, a wide range of music-related activities can be remotely supported by web-mediated technologies, thus unleashing unprecedented opportunities to foster access and diffusion of musical cultural heritage at artistic and commercial levels. Among those, Networked Music Performances (NMPs) involve multiple geographically displaced musicians performing together in real-time thanks to low-latency audio streaming over a telecommunication network. Networked Music Performance (NMP) systems are used in a variety of musical practices, including rehearsals, concerts, and pedagogy, and their need has become prominent during the recent COVID-19 pandemic [13].

The latency requirement in NMPs is subject to many studies. A maximum mouth-to-ear delay of 30 ms is suggested to allow synchronized and immersive interaction between musicians [16]. Since one of the key features of 5G communication systems is the capability of effectively support **low latency** services, there are significant expectations that the 5G system can become a key enabler for NMPs and, more generally, for Internet of Musical Things (IoMusT) scenarios.

In cellular networks, the Smart Musical Instruments (SMIs), often used in NMPs, can be seen as a completely new class of User Equipments (UEs). Thanks to cellular radio's plug-and-play concept, SMIs can achieve end-to-end connectivity with minimal configuration efforts both on the device side and on the network side, assuming to exploit the publicly available network infrastructure. The 5G system brings many novelty aspects in both the Radio Access Network (RAN) and the Core Network (CN) [15].

Reduced processing times at both the UEs and the base station, grant-free transmissions, antenna diversity, and multi-connectivity, make it possible for the 5G New Radio to meet the latency-reliability constraints, and to become a relevant enabler for networked musical interactions. As for the CN, the 5G service-based architecture seamlessly integrates Multi-access Edge Computing (MEC) platforms into the 5G core administrative domain: this makes it possible for the MEC host to interact with the CN, negotiate traffic and workload routing policies, as well as provide or exploit value-added services [15].

Because NMPs and IoMusT are highly time-critical applications, the MEC's role is as key as the RAN's or the transport network's: in fact, the MEC is a **perfect candidate location to receive and mix synchronous audio streams**, as well as to implement more advanced or machine learning-based functions. An example of the latter includes filling audio gaps originating from, e.g., bursts of wireless transmission errors, unrecoverable packet losses, and severely out-of-order packet deliveries [15].

The problem addressed in this thesis is precisely the **selection of the candidate site to host the server responsible for managing simultaneous communication among multiple performers participating in the same musical event (e.g., a concert, an ensemble rehearsal, or a music lesson)**. The objective of selecting the optimal candidate site is to minimize the average delay experienced by all users involved. In a 5G network scenario supporting a NMP, minimizing latency is essential, as it directly impacts the quality of the real-time interactive experience among users. Thus, this thesis proposes a heuristic algorithm that, at the instantiation of a new NMP session, focuses on identifying in real-time the most efficient server location to ensure low-latency communication, balancing the load while keeping delays to a minimum. This approach allows us to evaluate different configurations and site placements, optimizing the overall network performance for highly time-sensitive NMP applications. The performance of the proposed heuristic approach is assessed by benchmarking it against two Mixed Integer Linear Programs (MILPs) capable of identifying the optimal server placement over a predefined time horizon, assuming that the details of future NMP sessions (e.g., participants, starting time and duration) are known beforehand.

The remainder of the thesis is organized as follows: In *Chapter 2*, we introduce the state-of-the-art, providing numerical metrics regarding NMP performance and delay in 5G communications. *Chapter 3* explores theoretical concepts about the 5G infrastructure and Optical Networks, because they are the two pillars of our simulated communication scenarios. *Chapter 4* details the Python-based simulator that implements the proposed heuristic and the two MILP models adopted for comparing the outcomes. In *Chapter 5*, we present the experimental results and assess the performance of the heuristic approach by comparing its outputs to the optimal ones provided by the MILP models, such as latency and server load. Finally, *Chapter 6* concludes the thesis, discussing the implications of our findings.

Chapter 2

State-of-the-art

This chapter explores state-of-the-art research in NMP, with a specific focus on delay measurements in 5G environments and a review of recent academic advancements in latency reduction for NMP.

In NMP, latency is a critical factor that determines the overall performance quality. According to several studies, musicians require end-to-end latency below 25 milliseconds (ms) [11] [6] for seamless interaction. Delays beyond this threshold disrupt the natural timing and rhythm, making synchronous performance difficult. Additionally, latency consistency, often referred to as jitter, is essential to avoid rhythmic disturbances that could degrade musical quality. These latency requirements stem from physiological and cognitive constraints. The auditory system of musicians is highly sensitive to temporal deviations, meaning that a delayed signal can disrupt the perception of rhythm and harmony. Consequently, minimizing latency has been a focus in NMP research, with a particular emphasis on leveraging modern networking technologies like 5G to achieve these stringent performance metrics.

2.1 Measurements and Metrics in Networked Music Performance

The primary metrics of interest in NMP are end-to-end delay, jitter, and packet loss. Each metric plays a distinct role in determining the quality of the performance:

- *End-to-End Delay*: This refers to the time it takes for audio data to travel from one musician's device to another. Studies such as [14] have quantified acceptable delay ranges, suggesting that achieving **sub-25 ms latency** is ideal for professional-level performance.
- *Jitter*: Jitter measures the variability in packet arrival time, with high jitter causing irregular timing, impacting the musicians' ability to perform rhythmically. Jitter is particularly problematic in NMP due to its potential to disrupt musical synchronization. There are methods to mitigate jitter through adaptive buffering strategies, which aim to smooth out packet arrival variability.

- *Packet Loss*: Packet loss refers to the number of data packets lost during transmission, often resulting in missing audio samples and reduced sound quality. Packet loss can degrade the perceptual quality of the audio, and therefore, minimizing loss is essential. Even minimal packet loss can noticeably affect NMP quality, necessitating reliable network connections [6].

Collectively, these metrics form the foundation of NMP quality assessment. The continuous advancement of measurement tools and methodologies has enabled more accurate evaluations of NMP performances, paving the way for real-world deployments over 5G networks.

2.2 5G as an Enabler for Enhanced NMP

With its high data rates, low latency, and support for massive device connectivity, 5G has emerged as a changeful technology for NMP applications. Key 5G features that contribute to its suitability for NMP include Ultra-Reliable Low Latency Communications (URLLC), network slicing, and MEC. Together, these features support high-quality, low-latency transmission for interactive applications like NMP.

2.2.1 Delay and Latency Improvements with 5G

The latency improvements offered by 5G are significant compared to previous wireless networks. Typical 4G networks deliver latencies of around 40–60 ms, which are too high for NMP. In contrast, 5G networks with MEC can achieve latencies as low as 1–5 ms for optimized scenarios, thus satisfying the requirements needed for synchronous music collaboration. [15]. 5G can reduce end-to-end latency substantially when utilizing MEC, as it minimizes the physical distance data must travel to the core network.

In order to show in more detail the improvements brought by 5G in terms of latency, we adopt the research [15]: the used scenario represents a single-cell distribution of performers. Here, multiple IoMusT users connect locally to the same Next-generation NodeB (gNB) (5G Base Station). The gNB is located amidst six other cells, which generate interfering traffic.

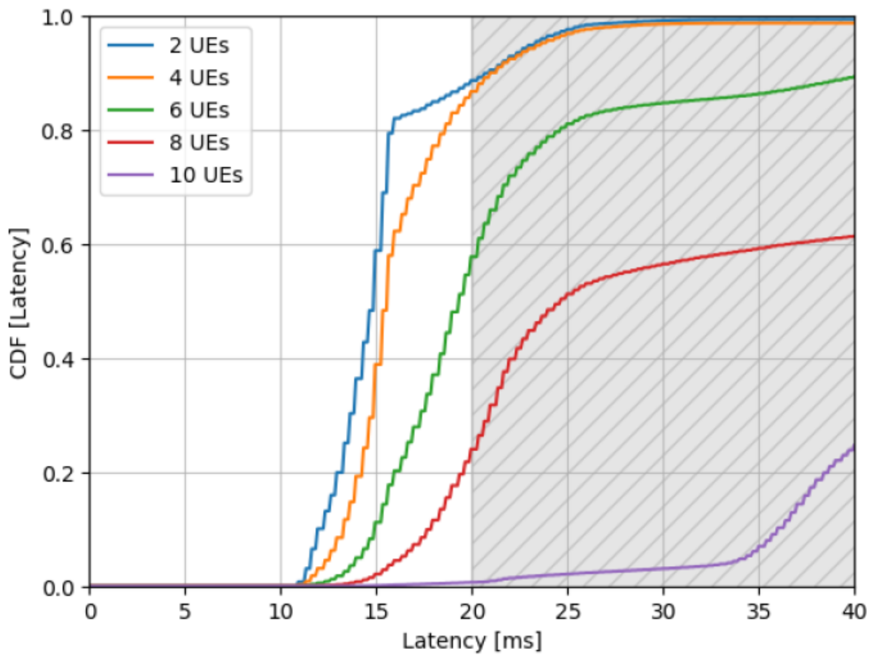


Figure 2.1: Packet delivery latency for 4G cellular architectures [15].

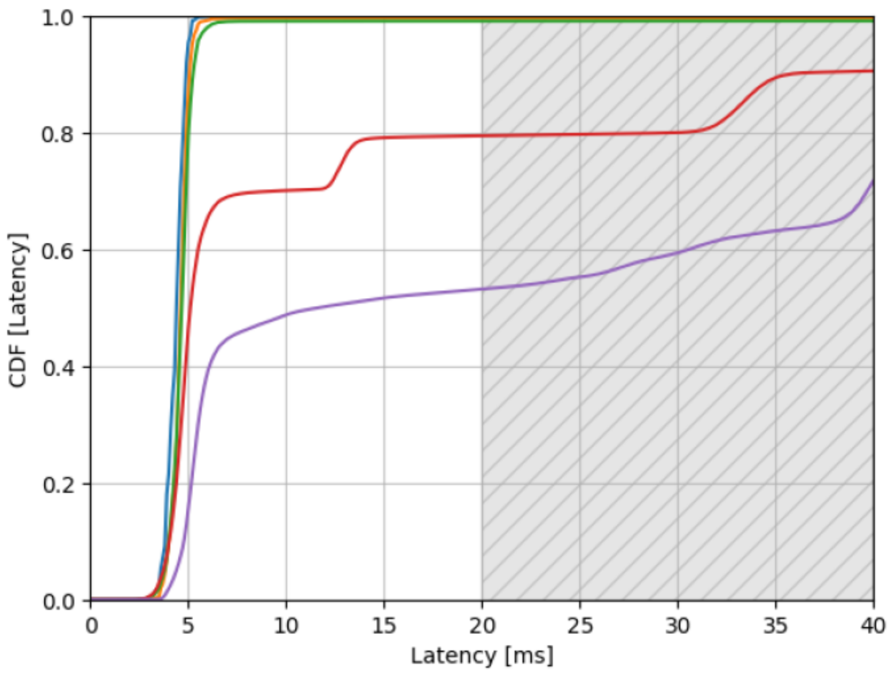


Figure 2.2: Packet delivery latency for 5G cellular architectures [15].

Figures 2.1 and 2.2 present the Cumulative Distribution Function (CDF) of the end-to-end packet delivery latency (i.e., the probability that the latency is less than the value indicated in the abscissa) for 4G and 5G cellular systems. Such latency value conveys the time required for a musical thing to send data to the MEC host and receive back a mixed audio stream. In 4G systems, latency exceeds 10 ms irrespective of the number of users in the network. This delay includes all components along the communication and processing chain, including the radio links and the MEC processing delay. In the simplest case with only two IoMusT devices, the maximum delay is already above the 20 ms threshold. As the number of users increases, the average and maximum delays grow, and the statistical dispersion of the delay also increases. For example, with 6 devices, 85% of the delivery delays exceed 15 ms, and the maximum delay is well above 20 ms. Conversely, the 5G architecture is more effective at delivering packets. Furthermore it reduces both the statistical dispersion of the latency (as seen from the steeper CDF curves) as well as both the minimum and maximum latency values: Less than 1% of the packets exceed 20 ms of delay for 6 users [15].

2.2.2 Network Slicing and Quality of Service (QoS) Management

Network slicing in 5G allows the creation of virtual networks tailored to specific applications. In NMP, where low latency and high reliability are paramount, network slicing enables a dedicated network slice that prioritizes the music performance traffic, providing an isolated and optimized pathway for data transmission. As highlighted in [8], network slicing could reduce congestion-related latency issues by isolating high-priority traffic from general network traffic, resulting in a stable and low-latency connection.

2.2.3 Multi-Access Edge Computing (MEC)

MEC is integral to achieving low latency in 5G networks. By processing data close to the user, MEC minimizes the time taken for audio data to travel between musicians in an NMP session. MEC's proximity to the end-users, combined with high processing power, reduces round-trip delays, thereby enhancing the quality of NMP.

2.3 Empirical Study on 5G-Enabled NMP

In paper [7], researchers focused on two typical 5G network architectures: a public pre-commercial 5G Non-Standalone (NSA) network, and a private 5G Standalone (SA) network with MEC infrastructure. They set up an NMP testbed in realistic radio access conditions (where all musical devices are closely co-located), collect performance metrics that help them assess the feasibility of each architecture for NMP, and perform a statistical analysis on their data. Their main aim was not that of comparing the two architectures, but that of quantifying their performances across latency and reliability metrics.

2.3.1 Experimental Setups

An end-to-end network (private or public) is typically composed of three elements:

1. **CN**: the central part of a network that provides services to users through the access network, and allows the transmission of IP packets to external networks such as the Internet.

2. **RAN**: the network infrastructure that includes radio base stations and bridges the connection between mobile radio network devices and the CN. To favor a smooth adoption of 5G technologies at least at the RAN side, 5G standards encompass two main configurations. The so-called SA configuration consists of the New Radio RAN connected to a natively 5G core network. Conversely, the NSA connects a 5G RAN to a 4G Evolved Packet Core.

3. **UEs**: any device directly used by an end user to communicate. This includes mobile smartphone appliances, communication systems embedded in low-power edge devices, as well as massive Internet of Things (IoT) communication devices.

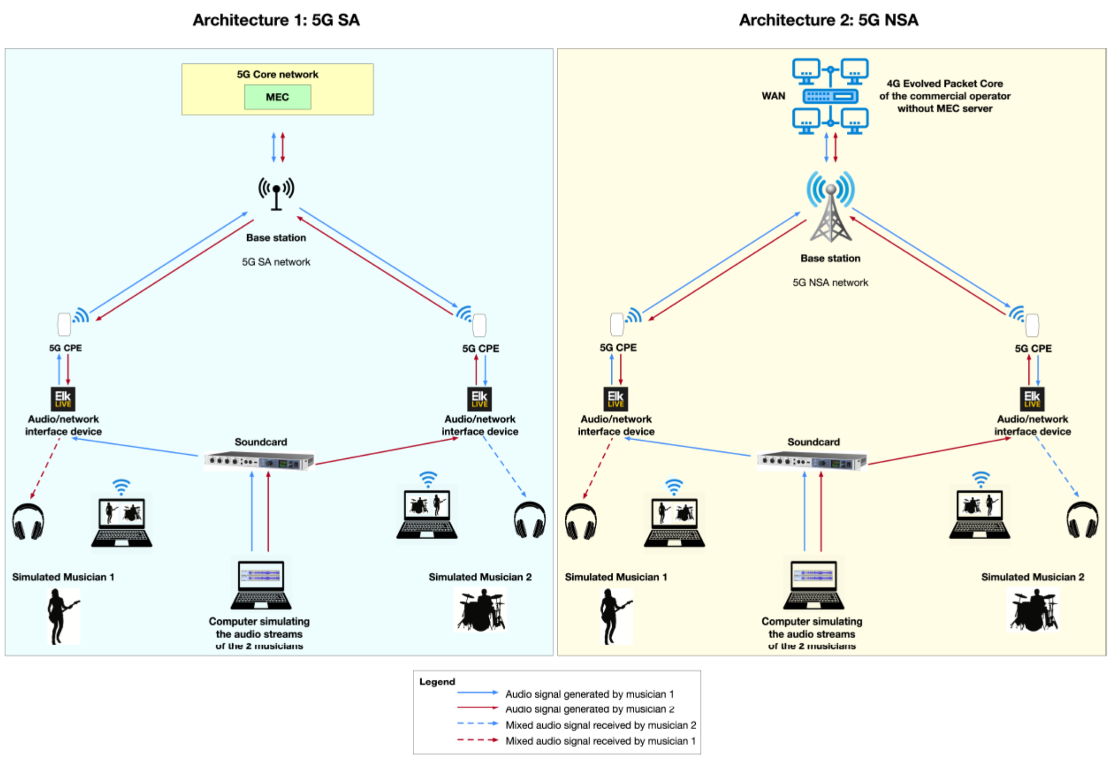

Architecture 1: 5G SA
Architecture 2: 5G NSA
5G Core network
MEC
WAN
4G Evolved Packet Core of the commercial operator without MEC server
Base station
5G SA network
5G NSA network
5G CPE
Audio/network interface device
Soundcard
Simulated Musician 1
Simulated Musician 2
Computer simulating the audio streams of the 2 musicians
Legend
Audio signal generated by musician 1
Audio signal generated by musician 2
Mixed audio signal received by musician 2
Mixed audio signal received by musician 1

Figure 2.3: Diagram and data flow of the two architectures deployed, the private 5G SA (left) and the pre-commercial public 5G NSA (right) [7].

The 5G architectures deployed are:

1. **Private 5G SA**: the base station was placed on the ceiling, roughly 3 m apart from two 5G UEs located on top of an office table (see Fig. 2.3, left). The two UEs acted, at the same time, as the sender and the receiver of digital audio packets. In this private 5G-SA network tests, the CN hardware was located in the same building as the base station, about 10 m apart, and connected via a fiber optic cable. Next to the base station, a MEC server was installed, which acted as a relay of the audio packets traffic between the peers.
2. **Public 5G NSA**: there are two key differences with respect to the private 5G SA architecture: (i) there is no reliance on a MEC server; and (ii) the traffic is conveyed from the base station to the commercial core network of the operator via a Wide-area Network (WAN) widespread on the Italian territory (see Fig. 2.3, right).

2.3.2 Experiments and Results

-	Mean	SD	Min	Max
Latency [ms]	21.87	0.32	21.02	22.96

Table 2.1: Results of the private 5G SA architecture

-	Mean	SD	Min	Max
Latency [ms]	74.26	146.74	22.01	830.95

Table 2.2: Results of the public 5G NSA architecture

The evaluation procedure was common to both setups and consisted in operating the NMP system for 10 minutes, during which the UEs continuously transmitted audio packets to each other. This enabled a rich set of measurements related to the performance of the network during the NMP. These results show that the public 5G NSA architecture exhibits higher baseline delays than the private 5G SA one, reaching 74 ms (see Tab. 2.2) against 22 ms for 5G SA deployment (see Tab. 2.1).

The 5G NSA architecture considered in the experiments proved insufficient to support NMP. As it can be noticed from Tab. 2.2, the performance of the network in terms of latency and reliability are well above the perceptual thresholds tolerable by musicians. The main reason behind this result is the large delay in end-to-end communications through the core network.

The 5G SA network deployment proved to be more suitable to NMP scenarios, mainly because of two reasons: the availability of edge server facilities, and the detachment from a WAN, where resources are necessarily shared among multiple flows with different priorities. These ingredients are fundamental to realize real-time musical interactions. Because concentrating processing and audio routing functions on MEC infrastructure makes it possible to minimize transit through the core network of the operator, the availability of a close MEC server emerges as a key ingredient of a successful 5G-enabled NMP scenario [7].

Chapter 3

Theoretical background

This chapter provides a theoretical background on the technologies underpinning the simulated communication scenarios used in this thesis: the 5G infrastructures as RAN and optical networks as the backend infrastructure. This discussion justifies the choice of these technologies by exploring their characteristics, capabilities, and suitability for NMP applications, particularly in terms of latency, reliability, and scalability.

3.1 Overview of 5G Infrastructure

The 5G infrastructure represents a significant evolution from previous wireless communication standards, offering capabilities uniquely suited to time-sensitive applications like NMP. The integration of technologies such as URLLC, MEC, and network slicing provides a solid foundation for real-time, low-latency audio streaming.

Ultra-Reliable Low Latency Communications (URLLC): One of the defining features of 5G is its support for URLLC, which enables end-to-end latencies as low as 1 ms under ideal conditions. URLLC ensures high reliability for critical applications through advanced scheduling algorithms, shorter transmission time intervals, and optimized retransmission techniques. According to [10], URLLC mechanisms in 5G enable the network to meet stringent latency and reliability requirements for applications such as industrial automation, autonomous vehicles, and NMPs.

The low-latency features of URLLC are critical for NMP applications, where delays exceeding 25–30 ms can disrupt musical synchronization, as discussed in Chapter 2. By ensuring predictable and minimal latency, 5G URLLC allows geographically distributed musicians to interact seamlessly, enhancing the feasibility of NMPs over public network infrastructures.

Multi-access Edge Computing (MEC): MEC is another fundamental element of 5G architectures, designed to reduce the round-trip latency by moving computation and data processing closer to end-users. The proximity of MEC servers to users minimizes the distance that data must travel, thus reducing end-to-end delays. MEC-enabled architectures in 5G are highly effective for latency-critical applications like augmented reality and online gaming. Similarly, in NMPs, MEC allows audio streams to be processed, mixed, and transmitted locally, ensuring that musicians experience minimal delays and jitter. Additionally, MEC enhances scalability by supporting dynamic workload distribution, which is critical for large-scale, multi-participant NMP scenarios.

Network Slicing: Network slicing allows 5G to allocate dedicated resources for specific applications or user groups, ensuring consistent performance even in congested networks. For NMPs, a network slice tailored for low-latency, high-reliability communication ensures that musical data packets receive the highest priority. This feature is particularly important in public 5G networks, where traffic from other applications can cause delays and packet loss.

Network slicing is a key aspect of 5G for mission-critical applications, enabling service differentiation and optimal resource allocation. In the context of NMPs, network slicing enables real-time resource allocation based on current network conditions. For example, during live performances, the slice assigned to NMP can dynamically adapt to fluctuations in bandwidth demand, maintaining uninterrupted audio quality [8].

3.2 Achieving High Performance in 5G: Exploring RAN and PON Architectures

The rollout of 5G networks has brought about a paradigm shift in telecommunications, offering unprecedented performance in terms of bandwidth, latency, and reliability. Achieving this level of performance requires an intricate and well-orchestrated infrastructure comprising both the wireless frontend and the underlying optical backbone. The wireless components of the 5G RAN provide seamless communication between user devices and the CN, while the Passive Optical Networks (PONs) ensure high-speed, low-latency data transmission across the network.

3.2.1 Frontend 5G Architecture and RAN

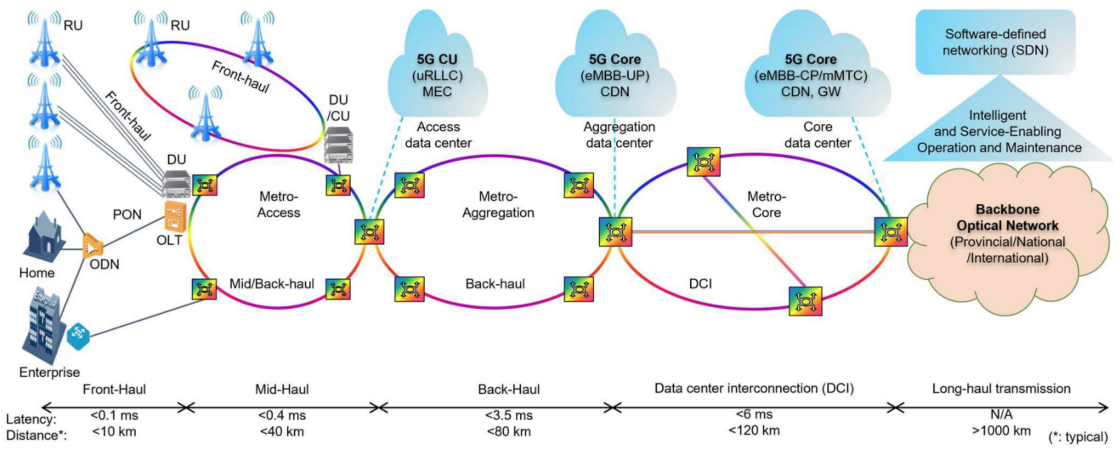


Figure 3.1: Illustration of an exemplary metro-area 5G C-RAN with front-haul, mid-haul, back-haul, and data center interconnection (DCI), as well as connection to backbone optical network. [12]

The 5G architecture, particularly its RAN, is designed to enable ultra-low latency, high bandwidth, and robust connectivity to meet the diverse demands of modern applications such as NMP. Central to this architecture (see Fig. 3.1) are several interconnected components, each playing a distinct role in enabling seamless communication:

- **Remote Units**: The Remote Unit (RU) serves as the endpoint of the RAN that interfaces directly with the UEs, such as mobile phones or IoT devices. The RU handles radio signal transmission and reception over the air interface. It performs tasks such as beamforming and signal modulation, leveraging technologies like Massive Multiple-Input Multiple-Output (MIMO) to support high-speed data transmission and robust connectivity. RUs are deployed close to the end-users, typically in cell towers or small cells, to enhance coverage and reduce latency [1] [5] [12].

- **Distributed Units**: The Distributed Unit (DU) is the intermediate layer of the 5G RAN, connecting the RUs to the Centralized Units (CUs). It performs real-time, latency-critical functions such as radio signal processing, scheduling, and resource allocation. By offloading these functions from the Centralized Unit (CU), the DU reduces the processing load at the central level while maintaining low latency for local tasks. DUs are often located in edge data centers or metro-access locations to minimize the distance to RUs [1] [5] [12].

- **Centralized Units**: The CU handles non-real-time tasks and more computationally intensive functions such as user mobility management, session control, and data aggregation. By centralizing these functions, the CU can optimize resource use and enable network slicing, which allows different applications (e.g. enhanced

Mobile Broadband (eMBB), URLLC, and massive Machine-Type Communications (mMTC)) to share the same physical infrastructure. CUs are typically hosted in regional or metro-access data centers [1] [5] [12].

- **5G Core Network**: The 5G CN represents the backbone of the 5G infrastructure. It is responsible for session management, authentication, policy control, and overall coordination of network resources. The 5G CN integrates various services such as the User Plan (eMBB-UP) for high-bandwidth applications and the Control Plan (eMBB-CP) for managing connections and signaling. Furthermore, the 5G CN supports Content Delivery Networks (CDNs) for optimized content distribution and mMTC for massive IoT connectivity [12].

- **Content Delivery Network**: The Content Delivery Network (CDN) layer in 5G ensures efficient content distribution by caching data closer to the end-users. This minimizes latency and reduces the load on core networks. CDNs are crucial for applications that require real-time streaming, such as virtual concerts or NMP, as they guarantee consistent performance even under high demand [12].

- **Centralized Radio Access Network**: Centralized Radio Access Network (C-RAN) is a critical aspect of the 5G architecture that centralizes the control of RUs, DUs, and CUs. By pooling resources in a centralized manner, C-RAN improves resource efficiency, scalability, and cost-effectiveness. It facilitates features like dynamic bandwidth allocation, precise synchronization, and network slicing, making it ideal for applications with strict latency and bandwidth requirements, such as NMP [12] [4].

- **Distributed Radio Access Network**: The Distributed Radio Access Network (DRAN) is an architectural framework for cellular networks, where the base station components are distributed and deployed closer to the end-users. Unlike the C-RAN, which consolidates baseband processing in centralized locations, DRAN retains baseband processing at each site. In a DRAN, each base station operates independently, which allows for localized processing and decision-making. While this architecture is simpler to deploy and provides resilience to failures at a central hub, it may suffer from limitations in scalability and coordination efficiency, particularly for applications requiring ultra-low latency or massive device connectivity [12] [4].

3.2.2 Passive Optical Network Components

A Passive Optical Network (PON) (see Fig. 3.2) forms the backbone of 5G infrastructure, connecting the network's endpoints with high-speed, fiber-based communication. PONs relie on passive components, eliminating the need for electrical power in the distribution network, which makes them energy-efficient and cost-effective [12] [2].

- **Optical Line Terminal**: The Optical Line Terminal (OLT) (see Fig. 3.2) is located at the service provider's data center. It acts as the endpoint of the PON on the provider's side. The OLT converts electrical signals into optical signals for transmission over the fiber network. Additionally, it manages traffic between the core network and the Optical Network Units (ONUs) or Optical Network Terminals (ONTs) at the user end. The OLT also dynamically allocates bandwidth and enforces Quality of Service (QoS) policies to ensure reliable service delivery.

- **Optical Network Units and Optical Network Terminals**: ONUs (see Fig. 3.2) and ONTs are the endpoint devices located at the user premises. They convert optical signals back into electrical signals for user devices such as routers or computers. The ONTs are typically deployed for single users, while the ONUs can serve multiple users in multi-dwelling units. These devices communicate with the OLT to receive and send data, ensuring seamless connectivity for end-users.
- **Optical Distribution Network**: The Optical Distribution Network (ODN) (see Fig. 3.2) is the passive infrastructure connecting the OLT to ONUs/ONTs. It consists of optical fibers, splitters, and connectors. Splitters divide the optical signal from the OLT into multiple signals to serve numerous users (like the Passive Optical Splitter), making PON highly scalable. The passive nature of the ODN reduces maintenance costs and ensures reliable service delivery.
- **Dense Wavelength Division Multiplexing** : Dense Wavelength Division Multiplexing (WDM) is a key technology in modern PONs that enables multiple optical signals to be transmitted simultaneously over a single fiber. By using different wavelengths for each signal, DWDM significantly increases the network's bandwidth capacity. This technology is critical for 5G applications, where high-throughput and low-latency communication are essential.

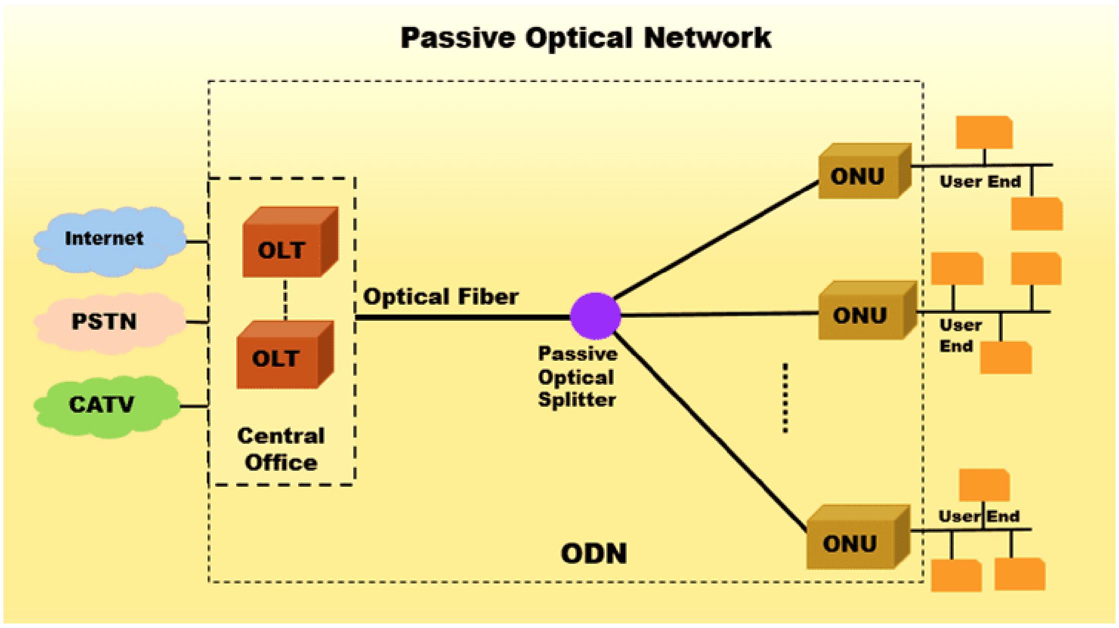


Figure 3.2: PON Components. PSTN: Public Switched Telephone Network; CATV: Community Antenna Television [3].

In summary, PONs provide the foundation for 5G's high-speed and low-latency connectivity. The integration of technologies like DWDM and sophisticated traffic management mechanisms ensures that the network can meet the rigorous demands of modern applications. Together with the RAN, PONs enable the seamless operation of 5G infrastructures, supporting a wide range of services and use cases.

3.3 Optical Networks as Backend Infrastructure

While 5G provides an efficient radio access network, the backend infrastructure also plays a crucial role in ensuring low latency and high reliability. Optical networks, with their data transmission speeds and minimal latency, are a natural choice for connecting MEC servers and core network elements in 5G-enabled NMP scenarios.

3.3.1 Characteristics of Optical Networks

Optical fiber networks provide data transmission speeds of up to 1 Tbps and latency under 1 millisecond over distances of tens of kilometers. These qualities make optical networks the preferred choice for supporting data transport in modern telecommunication systems. Low latency and high bandwidth of optical networks are essential for supporting 5G services such as URLLC and eMBB.

Optical networks are crucial for 5G networks as they provide connectivity between RUs, DUs, CUs, and the 5G CN (see Fig. 3.1). Additionally, they connect data centers for computation, storage, content generation, and routing. Modern optical networks must meet 5G demands for high bandwidth (enabled by wider Radio Frequency Spectrum and MIMO), low latency, precise synchronization, and network slicing to ensure quality of service and efficient resource use. Innovations in optical modulation techniques, such as Dense WDM, and advanced error correction algorithms further enhance their ability to handle high-throughput, low-latency data flows. This robustness is critical for applications like NMP, where uninterrupted data streams are essential [12].

3.3.2 Integrating 5G and Optical Network Technologies

Optical networks form the backbone of 5G infrastructure, connecting gNBs, MEC servers, and core network elements. The seamless integration of optical networks with 5G ensures that high-speed, low-latency data transport is maintained throughout the network. This integration is particularly important for NMPs, where end-to-end delay requirements are stringent. The combination of 5G RAN and optical fiber backhaul significantly reduces latency and jitter in multimedia applications, NMPs, gaming, etc,... This highlights the importance of optical networks in meeting the performance requirements of NMPs and other latency-sensitive use cases.

According to [9], 5G technology is being deployed globally to enhance connectivity for users, businesses, and IoT devices. Furthermore, 5G networks can use SA or NSA architectures, and service classes include eMBB, URLLC, and mMTC.

The 5G RAN is built around functional units (central unit, distributed unit, and radio unit) that can be deployed either in a DRAN or C-RAN setup, depending on site constraints and service needs. Different transport segments (fronthaul, midhaul, and backhaul) connect the RAN to the core network, with throughput demands ranging from 10 Gbps to 100 Gbps depending on configurations like MIMO or frequency range [9].

The Open RAN Alliance (O-RAN) is driving innovation toward interoperable, virtualized RAN systems, supported by Software-Defined Networking (SDN) and network abstraction. These advances enable better automation, slicing, and multi-operator scenarios. O-RAN focuses on creating open and standardized interfaces between the components of RAN, allowing interoperability between hardware and software from different vendors. Basically it decouples RAN software from hardware, allowing functions to run on general-purpose hardware in data centers or edge devices. The O-RAN represents a paradigm shift toward open, flexible, and intelligent RAN systems, ensuring that future networks are robust, efficient, and sustainable.

Fiber optics, particularly Point-to-Point (PtP) fiber, plays a critical role in supporting 5G's data transport, but resource constraints and existing fiber usage (e.g., Fiber-to-the-Home (FTTH)) require careful planning. Various optical transport solutions exist, including PtP, WDM (see 3.2.2), and Time Division Multiplexing-Passive Optical Networks (TDM-PON), to meet the stringent demands of 5G [9]:

- A **PtP optical network** directly connects two endpoints, typically using dedicated optical fibers. It provides an exclusive communication link between a source and a destination; each connection has its own fiber strand, which can be costly if many connections are required; supports very high bandwidth and low latency, making it suitable for critical or high-performance use cases.
- **TDM-PON** is a shared optical network architecture that uses time-division multiplexing to allocate bandwidth to multiple endpoints over a single fiber. Optical splitters divide a single fiber into multiple branches, connecting several users to a central point (OLT); each user is assigned a time slot for data transmission to avoid collisions, using Time Division Multiplexing (TDM). Suitable for connecting multiple small sites or homes with shared bandwidth.

3.4 Optical Networks and 5G combination

The combination of 5G for radio access and optical networks for backend infrastructure provides an ideal framework for supporting NMP applications because of the following reasons:

1. **Latency Minimization:**
    - URLLC and MEC reduce delays in the access and edge segments of the network.
    - Optical networks ensure near-zero latency in the transport and core segments.
    - Synchronization across 5G and optical elements eliminates timing mismatches that could otherwise introduce delays.

2. **Scalability:**
    - 5G supports large-scale connectivity for multiple NMP participants.
    - Optical networks' high bandwidth capacity handles the increased data loads generated by simultaneous audio streams.
    - Advanced multiplexing techniques in optical networks ensure efficient utilization of fiber resources, accommodating future scaling needs.
3. **Reliability:**
    - Network slicing in 5G ensures priority delivery of time-sensitive audio packets.
    - Optical networks offer high signal integrity and error resilience.
    - Redundant pathways in optical networks safeguard against disruptions, ensuring uninterrupted performance.

By combining these technologies, it is possible to meet the stringent requirements of NMP applications, enabling real-time, high-quality interactions among geographically distributed musicians. The next chapter will detail the Python-based simulator used to evaluate the performance of the proposed heuristic algorithm, as well as the MILP models, within the simulated 5G and optical network environment.

Chapter 4

Simulator and MILP Models

This chapter describes the Python-based simulation framework developed to optimize VM placement for audio mixing servers in NMP over 5G networks. To achieve low-latency, load-balanced server allocation, this framework simulates a 5G-enabled optical network environment where remote musical participants connect to audio mixing servers at specific network nodes. The objective of the simulation is to determine the optimal VM placement that meets latency constraints while maintaining balanced server loads. Two MILP models are integrated to serve as benchmarks for assessing the efficiency and latency performance of the heuristic approach embedded within the simulator. This chapter describes the implementation of these MILP models and their use in evaluating the simulator's effectiveness.

4.1 Design and Objectives of the Simulation Framework

4.1.1 Simulation Components

The simulator is designed to model an optical network functioning as a backend for 5G access, with multiple users connecting via 5G to participate in an NMP session (we called it **concert**). The key components of the simulator include:

- **Network Topology**: the Network Topology Module serves as the foundational component of the simulation framework, establishing the structure and properties of the network environment. This module is responsible for generating a randomized optical network topology, which provides the basis for simulating user locations, network nodes, inter-node connections, and link latencies. The topology is defined through a dedicated script that generates a JSON file, representing the entire network configuration. The final user of the simulator sets only the desired number of users and network nodes, while the module generates the topology autonomously based on these inputs. The network topology creation is a randomized process, which assigns each network node and user a random position within a defined area. These positional coordinates are fundamental for calculating the link latency between nodes, as they determine the physical distance each signal must travel. For an optical network, link latency is calculated using the formula:

$$latency = \frac{length}{2/3 \cdot c} \quad [s] \tag{4.1}$$

This formula is applied because light propagation in optical fiber is approximately two-thirds the speed of light in vacuum. As a result, physical distances between nodes directly impact latency, making node placement a critical factor in network performance.

The topology generation script incorporates two essential constraints:
- **No Direct User-to-User Connections**: Users in the simulation cannot establish direct connections with each other. Instead, they interact solely through network nodes, representing the real-world requirement of a centralized routing point for communications in optical networks.
- **Single Node Connection per User**: Each user is connected to exactly one network node, ensuring a clear and manageable user-to-network mapping.

The resulting JSON file includes:
- **Node and User Coordinates**: Precise spatial coordinates for each user and network node.
- **Node Interconnections**: Network nodes are linked to each other, with link latencies calculated based on their relative distances.
- **User-to-Node Assignments**: Each user is linked to a single, randomly assigned node, establishing initial connection points for subsequent simulation activities.

This topology module ensures that all user-server communications are routed through network nodes, with latency values grounded in realistic, distance-based calculations. This setup forms the structural basis for all subsequent steps in the simulation, as the latency and network node positions influence both the VM placement decisions and the end-to-end latency experienced by users during simulated Networked Music Performances.

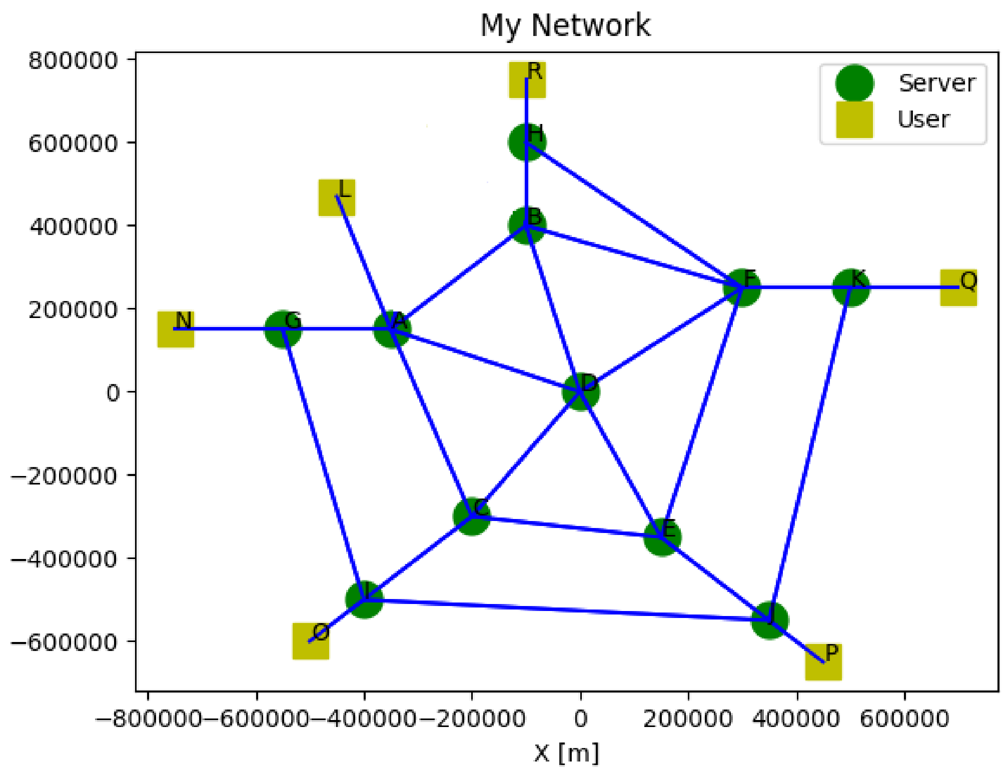
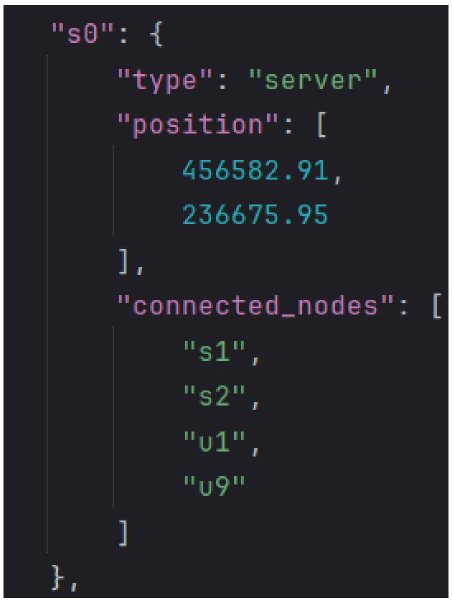


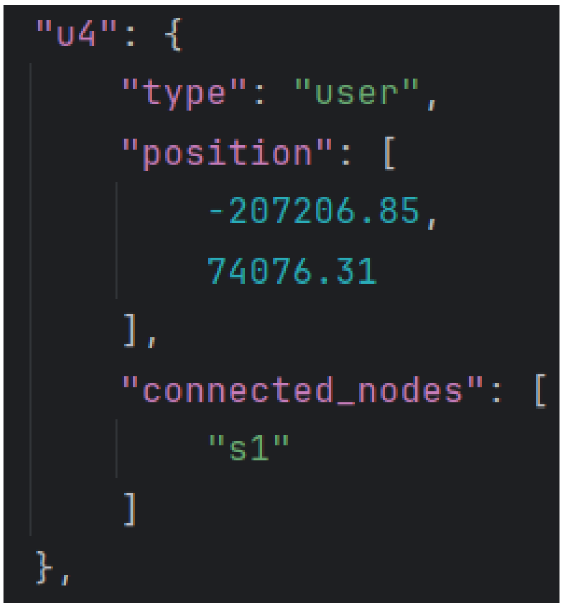
Figure 4.1: Example of Network Topology.

However, the final user can create his own JSON file based on the topology he/she wants to simulate, as long as it respects the naming convention (see Fig. 4.1.1) used by the simulator and the network topology constraints explained above (see Section 4.1.1).



```
"s0": {
    "type": "server",
    "position": [
        456582.91,
        236675.95
    ],
    "connected_nodes": [
        "s1",
        "s2",
        "u1",
        "u9"
    ]
},
```

Figure 4.2: Example of server definition in JSON file



```
"u4": {
    "type": "user",
    "position": [
        -207206.85,
        74076.31
    ],
    "connected_nodes": [
        "s1"
    ]
},
```

Figure 4.3: Example of user definition in JSON file

- **User Demand Model**: Generates user requests for a concert participation, defining which users will be involved, start and end time of the concert. the most important aspect to consider is that each user can only participate in one concert at a time.

- **VM Placement Decision**: It is the core of the simulator because it determines VM placement. The adopted algorithm follows these steps:

  1. Find all paths between each pair of users (musicians).
  2. Sort the paths, for each pair of users, according to their total latency (increasing order).
  3. When a new concert occurs, select the best path (lowest latency) for each sender-receiver pair among the set of participants.
  4. Compare the selected paths and extract the node/nodes in common.
  5. Among these nodes, select the one with the lowest load.
  6. If there are no nodes in common among all the best paths, start to looking for common nodes in other paths, progressively worse in terms of latency.
  7. If there is still no common node, the simulator rejects the concert request.

  The following pseudo-code goes into detail about each step:

```
# Input:
# - Users (U): List of user nodes (musicians)
# - Network Topology (T): Graph representation of the network
# - Latency (L): Dictionary of latencies for all paths
# - Concert Request (C): List of users participating in the new concert
# - Node Load (Load): Dictionary of load values for each network node

# Output:
# - Selected Node (N): The best network node to host the VM, or Reject if no suitable node exists

function VM_Placement_Decision(U, T, L, C, Load):
    # Step 1: Find all paths between each pair of users in C
    Paths = {}  # Dictionary to store paths for each user pair
    for user_a in C:
        for user_b in C:
            if user_a != user_b:
                Paths[(user_a, user_b)] = find_all_paths(T, user_a, user_b)

    # Step 2: Sort paths for each user pair by increasing latency
    for pair in Paths:
        Paths[pair].sort(key=lambda path: calculate_total_latency(path, L))

    # Step 3: Select the best (lowest latency) path for each sender-receiver pair
    Selected_Paths = []
    for pair in Paths:
        Selected_Paths.append(Paths[pair][0])  # Choose the path with the lowest latency

    # Step 4: Extract common nodes from the selected paths
    Common_Nodes = extract_common_nodes(Selected_Paths)

    # Step 5: If common nodes exist, select the one with the lowest load
    if Common_Nodes:
        N = find_node_with_lowest_load(Common_Nodes, Load)
        return N  # Return the selected node

    # Step 6: If no common nodes exist, search progressively in worse paths
    for k in range(1, len(Paths[list(Paths.keys())[0]])):  # Progressively look at worse paths
        New_Common_Nodes = []
        for pair in Paths:
            if k < len(Paths[pair]):
                New_Common_Nodes.extend(extract_common_nodes([Paths[pair][k]]))

        if New_Common_Nodes:
            N = find_node_with_lowest_load(New_Common_Nodes, Load)
            return N  # Return the selected node

    # Step 7: If no common node is found, reject the concert request
    return "Reject"

# Helper Functions
function find_all_paths(T, user_a, user_b):
    # Return all paths between user_a and user_b in the network topology T
    ...

function calculate_total_latency(path, L):
    # Calculate the total latency of a given path based on latency dictionary L
    ...

function extract_common_nodes(paths):
    # Extract nodes common to all given paths
    Common_Nodes = set(paths[0])
    for path in paths[1:]:
        Common_Nodes = Common_Nodes.intersection(set(path))
    return list(Common_Nodes)

function find_node_with_lowest_load(nodes, Load):
    # Return the node with the lowest load from the list of nodes
    return min(nodes, key=lambda node: Load[node])
```

To see the entire simulator code, refers to
https://github.com/angeloriccobene/VMs-placement-for-audio-mixing-servers-in-NMP-over-5G-networks

- **Performance Analysis Module**: Assesses server load, end-to-end latency, and computational timings, providing insights into the effectiveness of each VM placement approach.

4.1.2 Simulator's Parameters

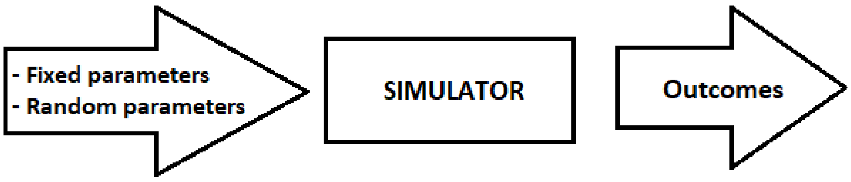


Figure 4.4: Simulator's Parameters.

The **Input Fixed Parameters** are:

- *Network Topology*: it includes the number of users, number of server nodes, latency in each link.
- *Concerts*: number of concert requests. Concerts can occur simultaneously.
- *Buffer Delay [s]*: introduced by the placed VM which acts as audio mixing server within the optical network.
- *Propagation Delay [s]*: each node crossed increases the propagation delay by a fixed delay amount δ due to queueing and processing operations.
- *Time Horizon [s]*: time frame within which the simulation takes place.
- *Musicians per Concert*: indicates the maximum number of participants per concert.
- *User Bit Rate [bps]*: it represents the amount of bits per second generated by a single user. It is calculated in the following way:

$$BitRate = SamplingFrequency \cdot BitDepth \cdot \#channels \quad [bps] \tag{4.2}$$

  - **Sampling Frequency**: Represents the number of samples of audio captured per second, measured in Hertz [Hz]. It determines the resolution of the audio in time. A higher sampling frequency captures more detail, leading to better audio fidelity.
  - **Bit Depth**: Represents the number of bits used to represent the amplitude (loudness) of each sample. It affects the dynamic range (difference between the loudest and softest sounds) and the precision of the audio. A 16-bit depth means each sample can represent 2^{16} possible amplitude values.

  - **Number of channels**: Refers to the number of separate audio signals recorded or played back. Determines the spatial aspects of sound (e.g. mono, stereo, etc,...).

- *Bandwidth Capacity per Server [bps]*: indicated the maximum bandwidth capacity for all the network nodes which can host the VMs. This capacity is expressed in terms of the maximum Bit Rate that can be handled at the same time by each node.
- *Concert Duration [s]*: indicates the maximum duration of a concert.
- *Delay Variation [%]*: this parameter introduces a controlled, random variation in latency during the simulation, expressed as a percentage increase over the baseline latency (see 4.1). It is designed to model real-world factors that contribute to variability in delay, such as the behavior of the RAN, buffering at intermediate nodes, and processing delays at the final node. These components are inherently subject to fluctuations caused by jitter, variations in channel access times in the wireless network, and queuing delays. By incorporating this variable, the simulation more accurately reflects the dynamic nature of end-to-end communication in optical networks integrated with wireless access points.
- *Tolerable User-Server Latency [s]*: indicates the maximum tolerable user-server latency in order to guarantee an adequate quality of the NMP session.

The **Input Random Parameters** are:

- *Number of Musicians*: each simulated musical event has a random number of participants.
- *Concert Time [s]*: within the fixed Time Horizon, the start and the end of each concert are randomly selected.

The **Outcomes** are:

- *Number of concerts*: how many concerts are served and the corresponding schedule (start and end time).
- *Network node*: which node has been selected by the algorithm, for each musical event, to host the VM that acts as audio mixing server.
- *Delay [s]* (included buffer and propagation delay):
  - total delay for each end-to-end transmission
  - maximum delay experienced by participants to any musical event
  - average delay experienced by participants to any musical event
- *Network node load [bps]*: for each network node in the topology, the total amount of load in terms of bandwidth consumption (the higher the number of musicians to manage, the greater the load) is calculated.

- *Execution time [s]*: simulation execution time. This value is important to be compared with the MILPs execution time to assess the performance of the simulator. To measure the execution time, the Python **time** module was utilized within both the simulator and the MILP models. The calculation involves recording the time at the beginning of the process (*start_time*) and the time when the process completes (*time.time()*). The elapsed execution time is then determined using the formula:

  $$execution\_time = time.time() - start\_time \quad [s] \tag{4.3}$$

  Here, *time.time()* retrieves the current time at the moment the function is called, while *start_time* corresponds to the timestamp recorded at the start of the operation. By applying this approach consistently across the simulator and the MILP models, it is possible to gather precise execution time values, facilitating a reliable comparison of their computational performance. The Execution time of the simulator corresponds to the duration of the Execution Step number 9 (see Fig 4.21). Instead for MILP1 and MILP2, it corresponds respectively to the duration of the Execution Step number 8.1 and 8.2 (see Fig 4.21).

4.1.3 Simulation Goals

The primary goals of the simulation framework are:

1. **Minimize End-to-End Latency**: Ensure real-timeNMP interactions by placing VMs at nodes that minimize user-to-server latency.
2. **Balance Server Loads**: Evenly distribute user demands across servers to prevent overload.
3. **Evaluate Computational Efficiency**: Compare the heuristic solution's computational speed and performance against two MILP models, revealing trade-offs between solution optimality and computational requirements.

4.2 MILP Models for VM Placement

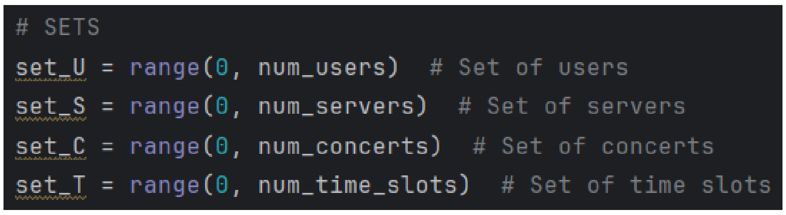
To benchmark the heuristic approach, two MILP models were developed using the **Gurobi** optimization library. These models solve the VM placement problem by focusing on either minimizing latency or balancing server load while meeting latency constraints.

4.2.1 MILP Model 1: Latency Minimization Model

Objective: This model prioritizes minimizing the maximum delay experienced by users connecting to audio mixing servers, ensuring that latency requirements for NMP are met across the network.

Sets: The model considers user-server connections, active NMP sessions, and latency per user-server pair across various time slots (see Fig 4.5).

- *Set* U: users
- *Set* S: server nodes
- *Set* C: musical events
- *Set* T: time horizon

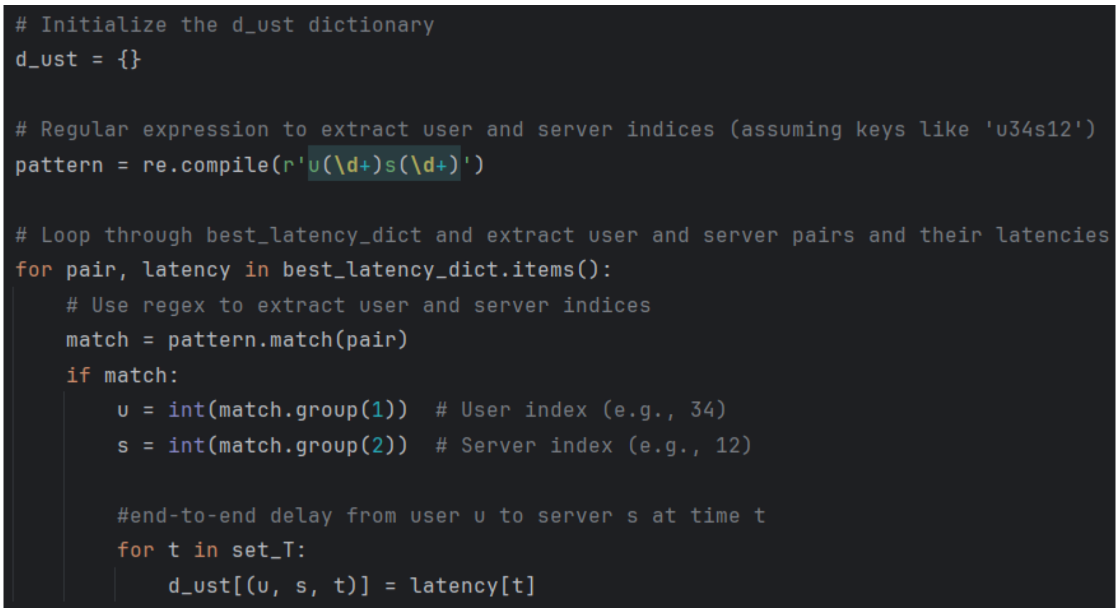


```
# SETS
set_U = range(0, num_users)  # Set of users
set_S = range(0, num_servers)  # Set of servers
set_C = range(0, num_concerts)  # Set of concerts
set_T = range(0, num_time_slots)  # Set of time slots
```

Figure 4.5: Sets initialization in MILP1.

Parameters

- d_{ust} $\forall u \in U, \forall s \in S, \forall t \in T$: matrix which represents the end-to-end delay from user u to server s at time t. It is constructed from network topology information (see Fig 4.6).

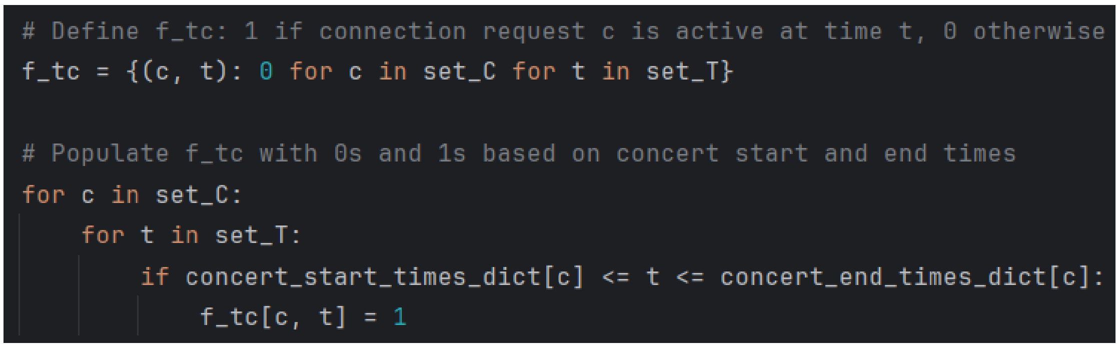


```
# Initialize the d_ust dictionary
d_ust = {}

# Regular expression to extract user and server indices (assuming keys like 'u34s12')
pattern = re.compile(r'u(\d+)s(\d+)')

# Loop through best_latency_dict and extract user and server pairs and their latencies
for pair, latency in best_latency_dict.items():
    # Use regex to extract user and server indices
    match = pattern.match(pair)
    if match:
        u = int(match.group(1))  # User index (e.g., 34)
        s = int(match.group(2))  # Server index (e.g., 12)

        #end-to-end delay from user u to server s at time t
        for t in set_T:
            d_ust[(u, s, t)] = latency[t]
```

Figure 4.6: Populate d_{ust} parameter in MILP1.

- M_s: maximum load capacity each server can handle.

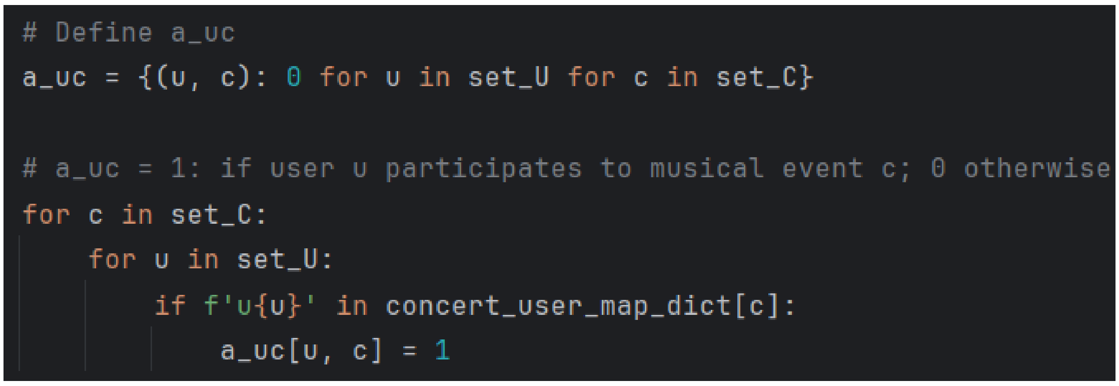
- f_{tc} $\forall t \in T, \forall c \in C$: binary matrix which represents if musical event c is active at time t, allowing concert start and end times to influence server load (see Fig 4.7).



```
# Define f_tc: 1 if connection request c is active at time t, 0 otherwise
f_tc = {(c, t): 0 for c in set_C for t in set_T}

# Populate f_tc with 0s and 1s based on concert start and end times
for c in set_C:
    for t in set_T:
        if concert_start_times_dict[c] <= t <= concert_end_times_dict[c]:
            f_tc[c, t] = 1
```

Figure 4.7: Populate f_{tc} parameter in MILP1.

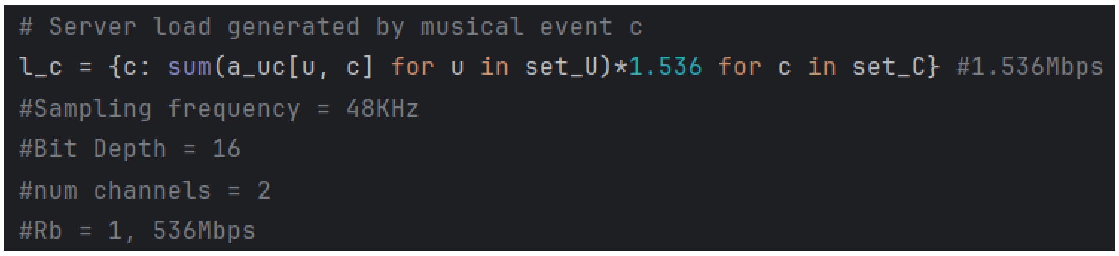
- a_{uc} $\forall u \in U, \forall c \in C$: binary matrix which represents if user u participates to musical event c (see Fig 4.8).



```
# Define a_uc
a_uc = {(u, c): 0 for u in set_U for c in set_C}

# a_uc = 1: if user u participates to musical event c; 0 otherwise
for c in set_C:
    for u in set_U:
        if f'u{u}' in concert_user_map_dict[c]:
            a_uc[u, c] = 1
```

Figure 4.8: Populate a_{uc} parameter in MILP1.

- l_c: total server node load for each musical event c (see Fig 4.9).

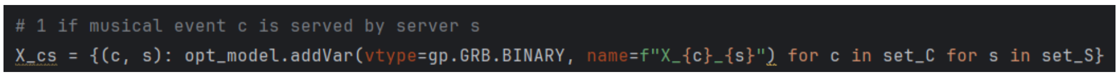


```
# Server load generated by musical event c
l_c = {c: sum(a_uc[u, c] for u in set_U)*1.536 for c in set_C} #1.536Mbps
#Sampling frequency = 48KHz
#Bit Depth = 16
#num channels = 2
#Rb = 1, 536Mbps
```

Figure 4.9: Populate l_c parameter according to bit rate formula (4.2) in MILP1.

Variables

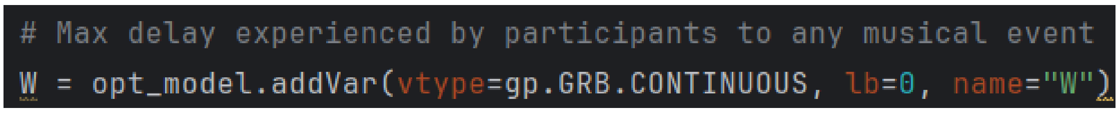
- X_{cs} $\forall c \in C, \forall s \in S$: **Binary Variable** which represents if musical event c is served by server node s (see Fig 4.10).



```
# 1 if musical event c is served by server s
X_cs = {(c, s): opt_model.addVar(vtype=gp.GRB.BINARY, name=f"X_{c}_{s}") for c in set_C for s in set_S}
```

Figure 4.10: Declare binary variable X_{cs} in MILP1.

- W: maximum delay experienced by musicians to any musical event, expressed as **Continuous Variable** (see Fig 4.11).



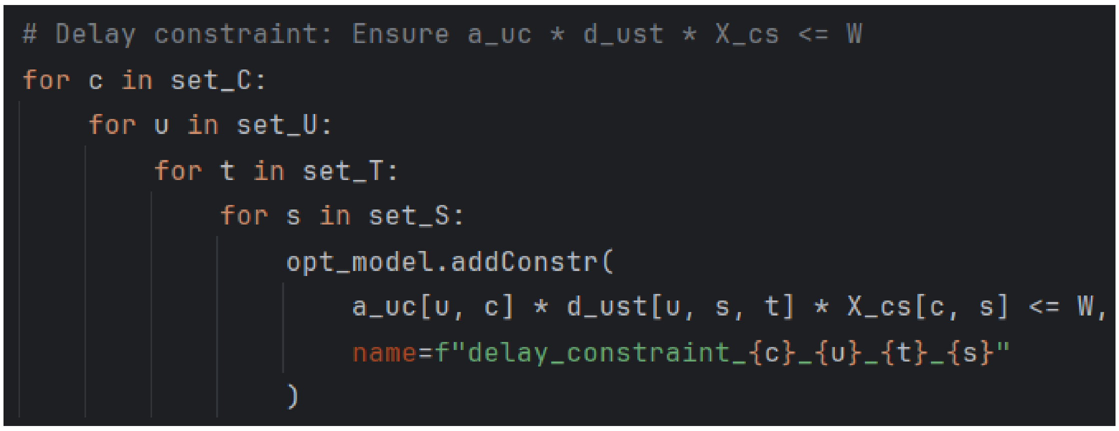
```
# Max delay experienced by participants to any musical event
W = opt_model.addVar(vtype=gp.GRB.CONTINUOUS, lb=0, name="W")
```

Figure 4.11: Declare continuous variable W in MILP1.

Constraints

- Coherence constraint: ensure that the maximum delay experienced by musicians cannot be exceeded (see Fig 4.12):

$$a_{uc} \cdot d_{ust} \cdot X_{cs} \leqslant W \quad \forall c \in C, \forall s \in S, \forall u \in U \tag{4.4}$$

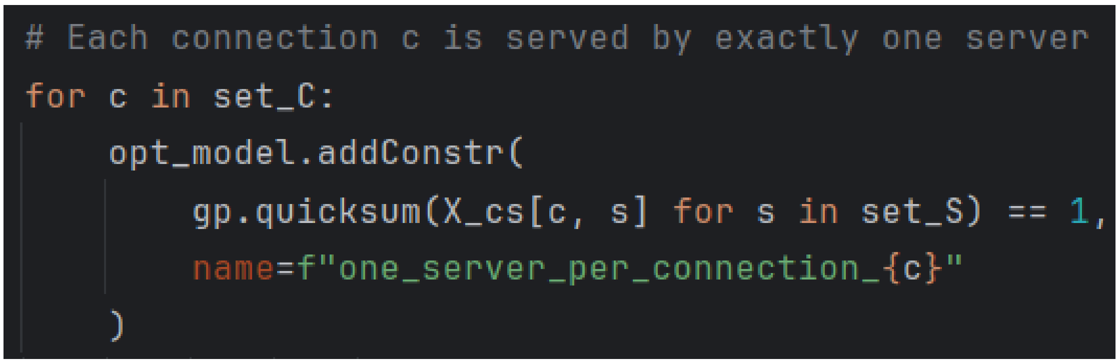


```
# Delay constraint: Ensure a_uc * d_ust * X_cs <= W
for c in set_C:
    for u in set_U:
        for t in set_T:
            for s in set_S:
                opt_model.addConstr(
                    a_uc[u, c] * d_ust[u, s, t] * X_cs[c, s] <= W,
                    name=f"delay_constraint_{c}_{u}_{t}_{s}"
                )
```

Figure 4.12: Coherence constraint in MILP1.

- Each musical event must be served exactly by 1 audio mixing server (see Fig 4.13):

$$\sum_{s \in S} X_{cs} = 1 \quad \forall c \in C \tag{4.5}$$

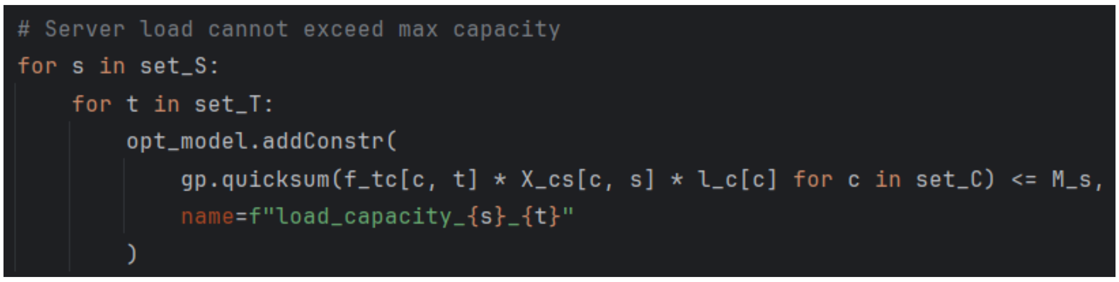


```
# Each connection c is served by exactly one server
for c in set_C:
    opt_model.addConstr(
        gp.quicksum(X_cs[c, s] for s in set_S) == 1,
        name=f"one_server_per_connection_{c}"
    )
```

Figure 4.13: One server per concert constraint in MILP1.

- Server node load cannot exceed maximum load capacity (see Fig 4.14):

$$\sum_{c \in C} f_{tc} X_{cs} l_c \leqslant M_s \quad \forall s \in S, \forall t \in T \tag{4.6}$$

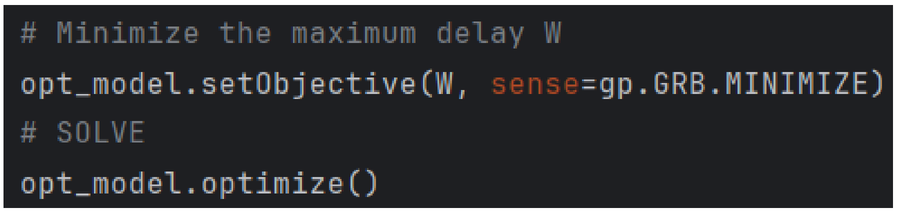


```
# Server load cannot exceed max capacity
for s in set_S:
    for t in set_T:
        opt_model.addConstr(
            gp.quicksum(f_tc[c, t] * X_cs[c, s] * l_c[c] for c in set_C) <= M_s,
            name=f"load_capacity_{s}_{t}"
        )
```

Figure 4.14: Server load capacity constraint in MILP1.

Objective function: the objective function seeks to minimize the maximum end-to-end delay occurs during the NMP sessions:

$$Min \;\; W \tag{4.7}$$



```
# Minimize the maximum delay W
opt_model.setObjective(W, sense=gp.GRB.MINIMIZE)
# SOLVE
opt_model.optimize()
```

Figure 4.15: MILP1's objective function.

This formulation balances latency across connections, offering a comprehensive solution to the latency minimization problem (see Fig 4.15).

4.2.2 MILP Model 2: Load Balancing Model

Objective: The second model focuses on minimizing server load by balancing the load across servers while meeting strict latency requirements.

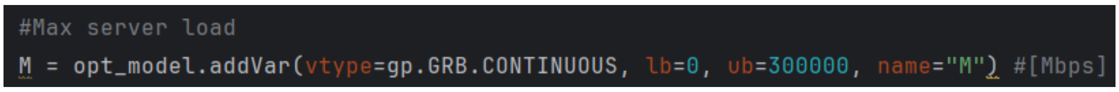
Sets: inherits the sets from Model 1.

Parameters: inherits the parameters from Model 1 and adds:

- D: represents the upper latency limit for any user-server connection.

Variables: inherits the parameters from Model 1 and adds:

- M: **Continuous Variable** representing the maximum load across all servers (see Fig 4.16).



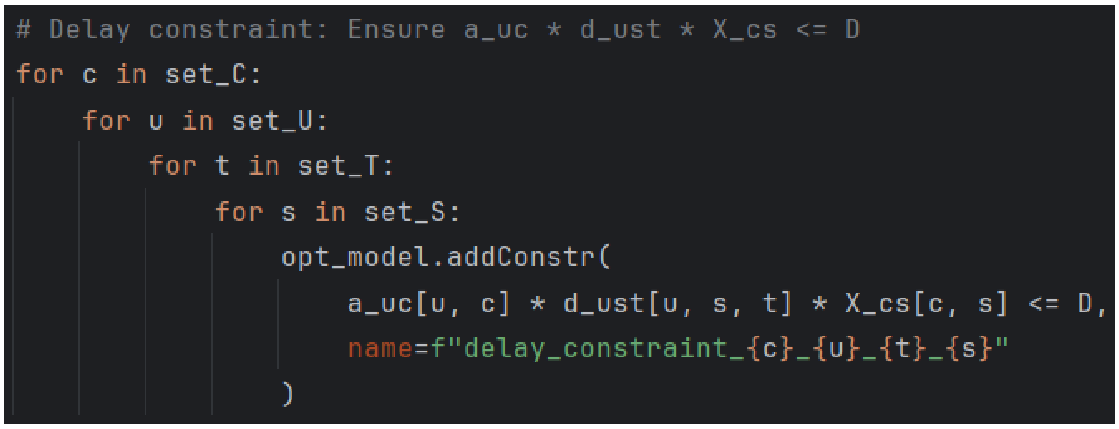
```
#Max server load
M = opt_model.addVar(vtype=gp.GRB.CONTINUOUS, lb=0, ub=300000, name="M") #[Mbps]
```

Figure 4.16: Declare continuous variable M in MILP2.

Constraints

- Latency constraint: maintains latency for each user-server connection below the threshold D (see Fig 4.17):

$$a_{uc} \cdot d_{ust} \cdot X_{cs} \leqslant D \quad \forall c \in C, \forall s \in S, \forall u \in U \tag{4.8}$$

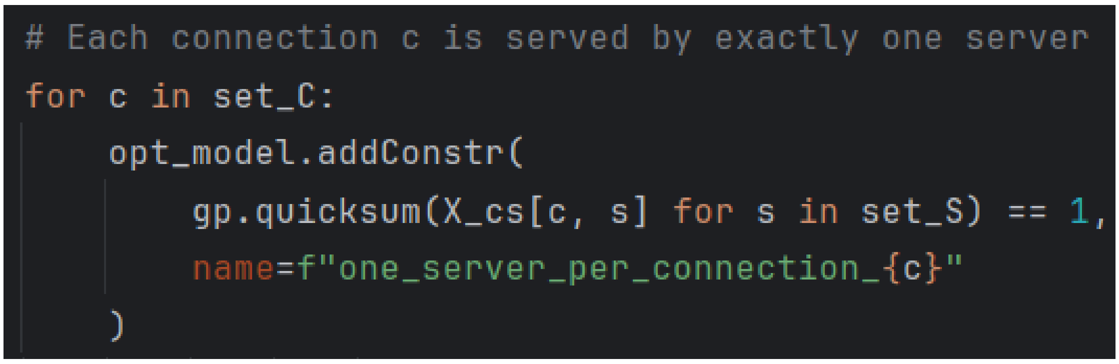


```
# Delay constraint: Ensure a_uc * d_ust * X_cs <= D
for c in set_C:
    for u in set_U:
        for t in set_T:
            for s in set_S:
                opt_model.addConstr(
                    a_uc[u, c] * d_ust[u, s, t] * X_cs[c, s] <= D,
                    name=f"delay_constraint_{c}_{u}_{t}_{s}"
                )
```

Figure 4.17: Latency constraint in MILP2.

- Each musical event must be served exactly by 1 audio mixing server (see Fig 4.18):

$$\sum_{s \in S} X_{cs} = 1 \quad \forall c \in C \tag{4.9}$$

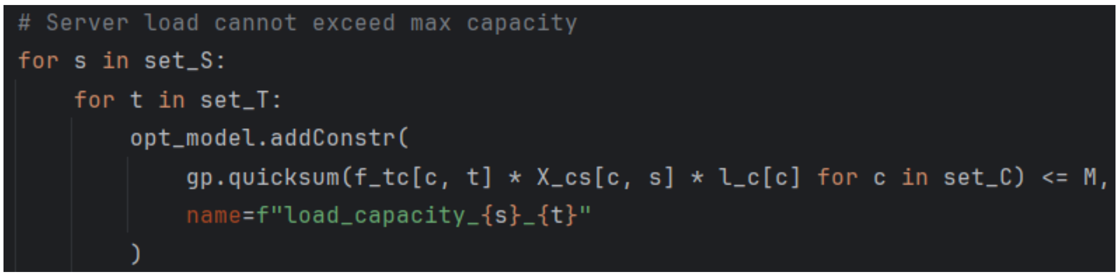


```
# Each connection c is served by exactly one server
for c in set_C:
    opt_model.addConstr(
        gp.quicksum(X_cs[c, s] for s in set_S) == 1,
        name=f"one_server_per_connection_{c}"
    )
```

Figure 4.18: One server per concert constraint in MILP2.

- Server Capacity Constraint: restricts load on each server to a maximum of M for each time slot (see Fig 4.19):

$$\sum_{c \in C} f_{tc} X_{cs} l_c \leqslant M \quad \forall s \in S, \forall t \in T \tag{4.10}$$

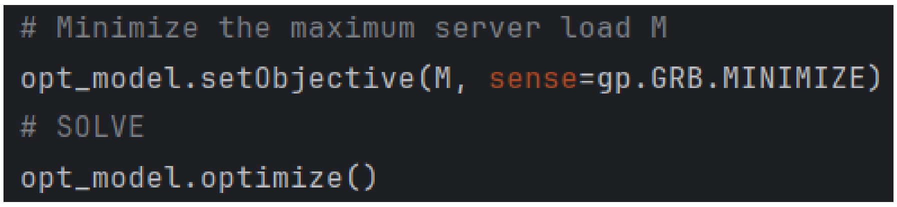


```
# Server load cannot exceed max capacity
for s in set_S:
    for t in set_T:
        opt_model.addConstr(
            gp.quicksum(f_tc[c, t] * X_cs[c, s] * l_c[c] for c in set_C) <= M,
            name=f"load_capacity_{s}_{t}"
        )
```

Figure 4.19: Server capacity constraint in MILP2.

Objective function: the objective function seeks to minimize the maximum server load across all servers:

$$Min \;\; M \tag{4.11}$$



```
# Minimize the maximum server load M
opt_model.setObjective(M, sense=gp.GRB.MINIMIZE)
# SOLVE
opt_model.optimize()
```

Figure 4.20: MILP2's objective function.

This model is useful in scenarios where load balancing is critical, ensuring no server is overburdened while still adhering to the latency constraints (see Fig 4.20).

4.3 Cooperation of the Simulator and MILP Models

The simulator and the two MILP models do not cooperate directly but they are initialized with the same input data to operate on identical network scenarios. This ensures a fair comparison of their performance under consistent conditions. Both the simulator and the models aim to address the NMP scenario by determining the placement of audio mixing servers while minimizing end-to-end latency and server load. By using the same input parameters, they evaluate the same network topology and event configurations, providing insights into routing efficiency and resource allocation strategies. Below is a step-by-step description of the process, from the creation of the network topology to the final execution and analysis of the simulation outputs.

Detailed Execution Steps

1. *Network Topology Creation*: The process begins with the creation of a network topology JSON file. The user specifies the desired number of users and network nodes, and the dedicated script randomly generates their positions. This file captures:

   - The spatial positions of users and network nodes, which are fundamental for determining latency values based on the link distance.
   - Connectivity rules, including the restriction that users cannot communicate directly with one another and can only connect to a single network node.

   The generated JSON file is then imported into the simulator, which uses this information to define the network structure.

2. *Setting Fixed Parameters*: Once the topology file is imported, the fixed input parameters are set within the simulator, including specifications for bandwidth requirements, server capacities, and latency thresholds. These parameters ensure that the system operates under realistic conditions.

3. *Simulation Start: Musical Event Generation*: When the final user clicks the RUN button, the simulation generates a random number of musical events, each with a unique set of participating musicians, a start time, and an end time. This randomization introduces a dynamic element to the simulation, mimicking real-world variability in event timing and participant configuration.

4. *Network Topology Reconstruction*: The simulator reconstructs the network topology as objects, defining nodes (both user and server nodes) and lines based on the JSON file data. It must establish connectivity, identifying how nodes are linked and defining paths for possible user-to-user communication.

5. *Path Calculation for User Pairs*: The simulator calculates all possible communication paths between each user pair in the network. Paths are evaluated based on link lengths, which are obtained from the node positions in the JSON file, allowing the simulator to calculate latencies for each path using the optical link latency formula (see 4.1). The resulting paths are sorted in ascending order of latency, providing each user pair with a prioritized list of paths from the lowest to the highest latency.

6. *Latency Calculation for User-Server Connections*: The simulator creates latency lists for each user-server pair, using the same formula and path-sorting approach as in the user-pair calculation. These lists allow the simulator to quickly identify the best possible server for each user based on latency, which is crucial for optimizing user-server communication during performances.

7. *Latency Variation Over Time*: To simulate real-world conditions, latency value (the lowest one selected) for each user-server pair fluctuates randomly over the time horizon of the simulation. The latency for the best path is modified by up to 30% in random increments, adding variability that more accurately reflects dynamic network conditions.

8. *Triggering MILP Models*: With all necessary data generated and structured, the simulator triggers MILP1 and MILP2 sequentially:

   - **MILP1**: The first model receives latency values for each user-server pair, details of each musical event (start/end times and participating users), and network parameters. It calculates the optimal maximum end-to-end latency across all events, minimizing this value as its primary objective. MILP1 also returns the average delay for comparison with the simulator's output.
   - **MILP2**: Using the same parameters as MILP1, MILP2 calculates the optimal maximum server load across all events, aiming to minimize this value as its objective.

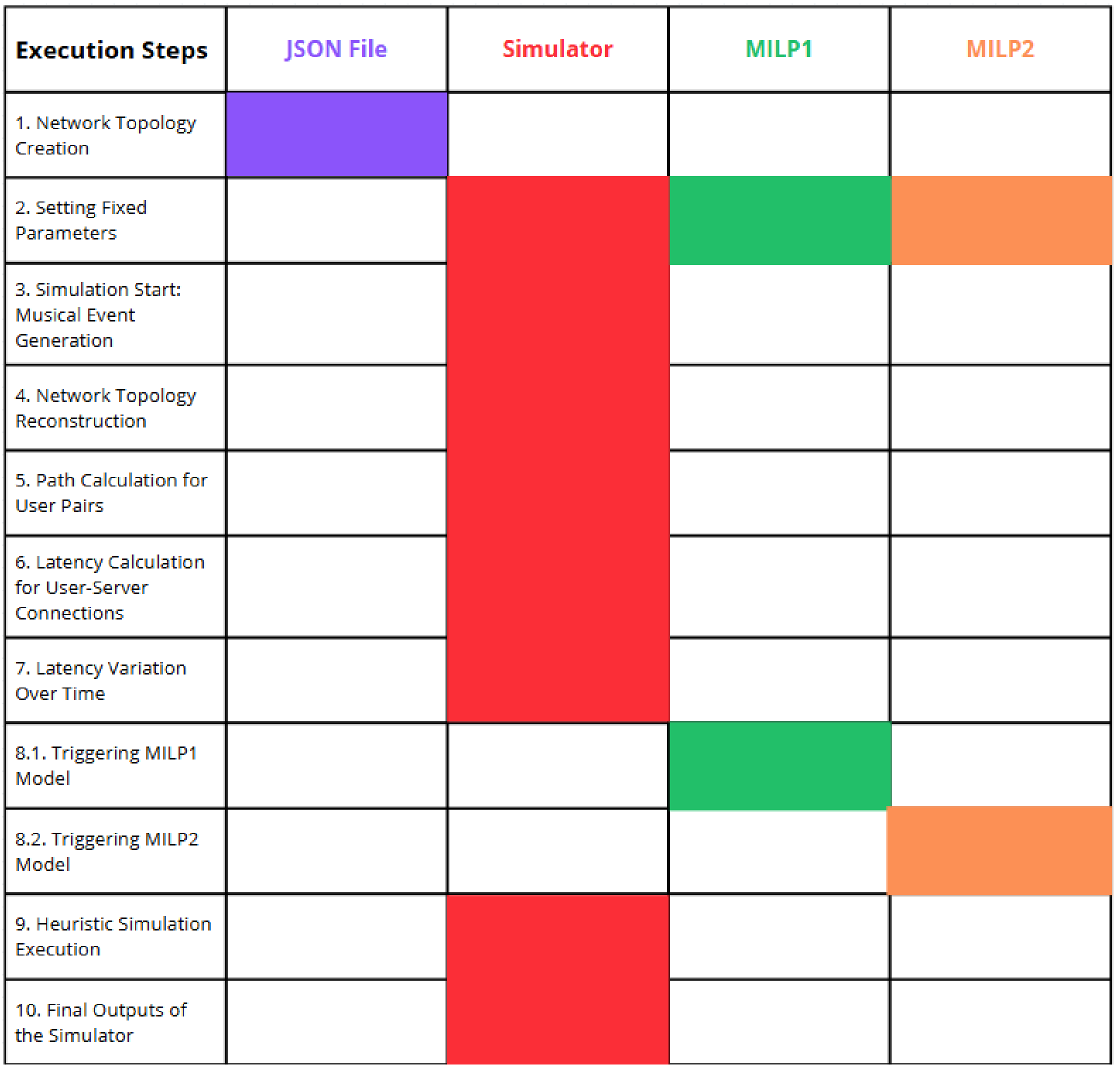
   Both models also return execution times, enabling comparative performance analysis between the heuristic simulation and the MILP-based optimizations.

9. *Heuristic Simulation Execution*: With the optimal values from MILP models as benchmarks, the simulator then proceeds with its heuristic-based approach to server selection and routing:

   - For each new musical event, the simulator identifies the lowest-latency paths for each user pair in the event.
   - It examines these paths to find common nodes, representing potential mixing server candidates.
   - Among common nodes, the simulator selects the least loaded node as the optimal server for the event.
   - If no common node exists, the simulator iterates through progressively higher-latency paths until a common node is found. If no such node can be identified, the event request is rejected, as no feasible solution meets the latency and load requirements.

10. *Final Outputs of the Simulator*: The simulator's heuristic execution yields key outputs showed above (4.1.2):

This step-by-step interaction between the simulator and the MILP models demonstrates a layered approach to solving the placement and routing problem in NMP. The simulator's heuristic solution provides a scalable, faster alternative, while the MILP models serve as benchmarks, providing optimal solutions for delay and load metrics to ensure the heuristic's results remain within an acceptable range of optimal performance.



Execution Steps	**JSON File**	**Simulator**	**MILP1**	**MILP2**
1. Network Topology Creation	🟪			
2. Setting Fixed Parameters		🟥	🟩	🟧
3. Simulation Start: Musical Event Generation		🟥		
4. Network Topology Reconstruction		🟥		
5. Path Calculation for User Pairs		🟥		
6. Latency Calculation for User-Server Connections		🟥		
7. Latency Variation Over Time		🟥		
8.1. Triggering MILP1 Model			🟩	
8.2. Triggering MILP2 Model				🟧
9. Heuristic Simulation Execution		🟥		
10. Final Outputs of the Simulator		🟥		

Figure 4.21: Simulation timeline.

Chapter 5

Experiments and Results

This chapter presents the results of an extensive performance evaluation conducted on the heuristic-based network simulator and the two optimization-based MILP models (MILP1 and MILP2). The experiments are structured to investigate the behavior and efficiency of both approaches under varying conditions, highlighting their applicability in allocating VMs to host audio mixing servers for real-time musical performances. By comparing the results obtained from the simulator and the MILP models, this chapter provides insights into the trade-offs between computational efficiency and optimization accuracy.

The analysis is divided into two distinct scenarios, each designed to isolate specific parameters of the network topology and system operation:

1. **Varying the Number of Server Nodes**: In this scenario, the number of servers in the network topology is gradually increased while keeping all other parameters constant. This experiment assesses how the availability of additional servers impacts the performance in terms of:

   - *Average Delay*: The mean end-to-end delay experienced by participants during musical events (simulator vs MILP1).
   - *Maximum Delay*: The worst-case end-to-end delay during any musical event (simulator vs MILP1).
   - *Maximum Server Load*: The highest bandwidth load observed on any server node in the network (simulator vs MILP2).
   - *Execution Time*: The computational time required by the simulator and the two MILP models to generate results.

   This analysis provides insights into the scalability of the network and the ability of the heuristic simulator to balance delays and server loads as more resources become available.

2. **Varying the Number of Musical Events**: In this scenario, the number of musical events occurring during the simulation is progressively increased while keeping

the network topology and all other parameters unchanged. This experiment evaluates the system's capability to handle higher loads and denser user participation, observing the same performance metrics as in Scenario 1:

- *Average Delay*: Evaluating how additional musical events influence latency for participants (simulator vs MILP1).
- *Maximum Delay*: Measuring the highest delay encountered in scenarios with higher event density (simulator vs MILP1).
- *Maximum Server Load*: Quantifying the strain on servers as the number of simultaneous events increases (simulator vs MILP2).
- *Execution Time*: Comparing the computational time required to process increased complexity in both heuristic and optimization approaches.

The purpose of these two scenarios is to **analyze the system's behavior under contrasting conditions**: one that increases network resources (servers) and another that increases network demands (musical events). By isolating these variables, we can better understand the relative strengths and limitations of the heuristic and MILP-based approaches, both in terms of performance optimization and computational feasibility.

The results presented in this chapter are expressed numerically and graphically, offering a clear comparison between the simulator and the MILP models. Through this analysis, we aim to identify key patterns, scalability considerations, and performance bottlenecks, ultimately guiding decisions about the most suitable method for managing latency-sensitive applications in optical networks.

5.1 Varying the Number of Server Nodes

This scenario involves progressively increasing the number of servers in the network topology while maintaining all other parameters unchanged. The goal of this experiment is to evaluate how the presence of additional servers influences the performance in terms of *average delay*, *maximum delay*, *maximum server load* and *execution time*.

The parameters that remain constant in this scenario are shown in Tab. 5.1:

Number of musicians	50
Number of musical events	30
Time horizon [min]	500
Maximum number of participants per musical event	8
User bit rate [Mbps]	1,536
Maximum musical event duration [min]	120
Maximum bandwidth capacity per server node [Gbps]	50
Queueing and Processing delay per node [ms]	3
Buffer delay [ms]	5
Delay variation [%]	30
Maximum tolerable user-server latency [ms]	1000
Propagation speed in the optical fiber [Km/s]	199862

Table 5.1: Scenario 1: constant parameters meanwhile varying the number of server nodes

5.1.1 Analysis and Explanation of Results: Simulator vs MILP1 (Delay)

The provided results (see Fig. 5.1, 5.2) illustrate the relationship between the number of server nodes in the network topology and the average and maximum delay (in seconds) experienced during the simulation. Both figures demonstrate that the increase in server nodes facilitates better VM placement, leading to reduced delays, which is critical for NMP systems. The error bars in the plot represent the range of values obtained across the 10 runs, specifically indicating the minimum and maximum values that the corresponding metric assumed during these simulations, thereby highlighting the variability of the simulation process. The key aspects of this comparison are outlined below:

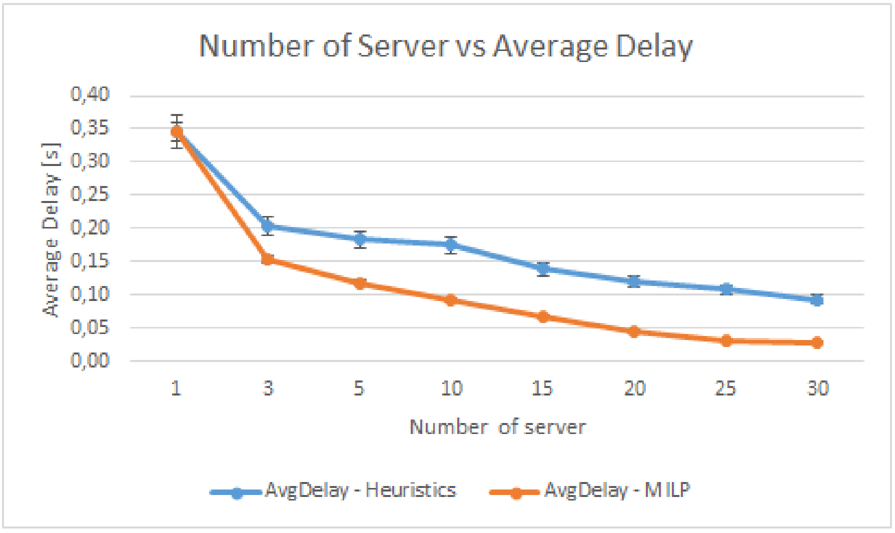


Figure 5.1: Varying the number of server nodes: average delay comparison.

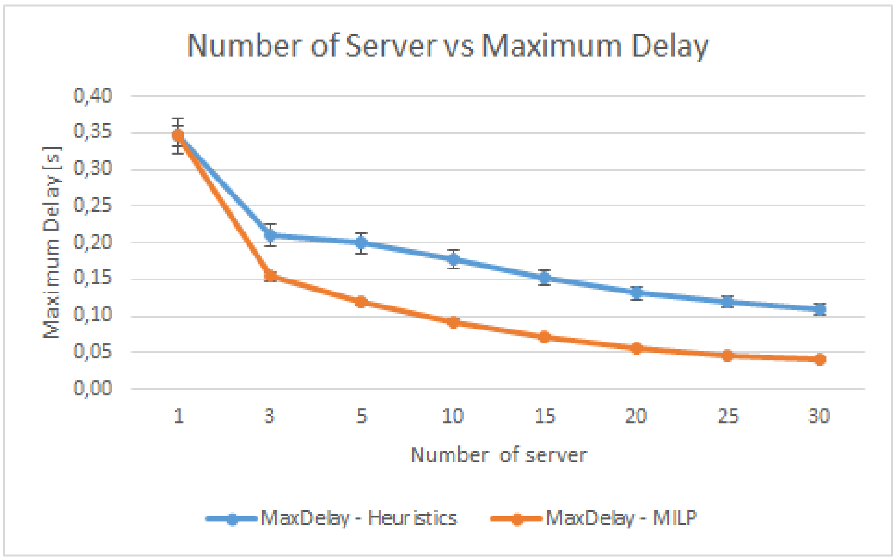


Figure 5.2: Varying the number of server nodes: maximum delay comparison.

1. **General Trends**

   - *Decreasing delay with increasing servers*: In both average and maximum delay metrics, the delay consistently decreases as the number of servers increases. This trend is intuitive; more servers enable a finer-grained allocation of VMs, thereby optimizing the proximity between the musicians and the servers.
   - *MILP1 Superiority*: The MILP1 approach consistently outperforms the heuristic method across all configurations. This is evident in both the average and maximum delay figures, where MILP1 achieves lower delay values for every server count.
   - *Delay Magnitudes*: As expected, the maximum delay values are always higher than the average delay values for both approaches. This is because the maximum delay measures the worst-case scenario, whereas the average delay is a broader representation of the overall delay distribution across all musicians and concerts.
   - *Variability*: The heuristic simulator exhibits greater variability in average and maximum delay values across the 10 simulation runs (as shown by the wider ranges in the plots), reflecting its reliance on a heuristic approach that is subject to non-deterministic behaviors. On the other hand, MILP1 results are more consistent, with narrower bars, due to its deterministic optimization process.

2. **Detailed Analysis of Results**

   - *One Server configuration*: At just one server, both approaches face significant limitations in VM placement flexibility, resulting in high delays.
   - *Intermediate Server counts (3–15 Servers)*: With 3 servers, the heuristic approach shows average and maximum delays of 0,2030 s and 0,2110 s, respectively. The MILP1 model, however, reduces these to 0,152 s and 0,1537 s. This

difference highlights the MILP1 model's capability to balance delay across the network effectively. For larger configurations, such as 10 or 15 servers, the MILP1 continues to outperform the heuristic approach, achieving delays below 0,1 s for both metrics.

- *High Server counts (20–30 Servers)*: With 20 servers, the heuristic approach's average delay is 0,1199 s, while the MILP1 achieves a significantly lower value of 0,0449 s. Similarly, the maximum delay values are 0,1309 s (heuristic) and 0,0562 s (MILP1). At 30 servers, the delay reduction is even more pronounced, with the MILP1 model reducing the average delay to 0,0289 s and the maximum delay to 0,0403 s, compared to 0,0930 s and 0,1080 s, respectively, for the heuristic method.

3. **Performance Gap Analysis**

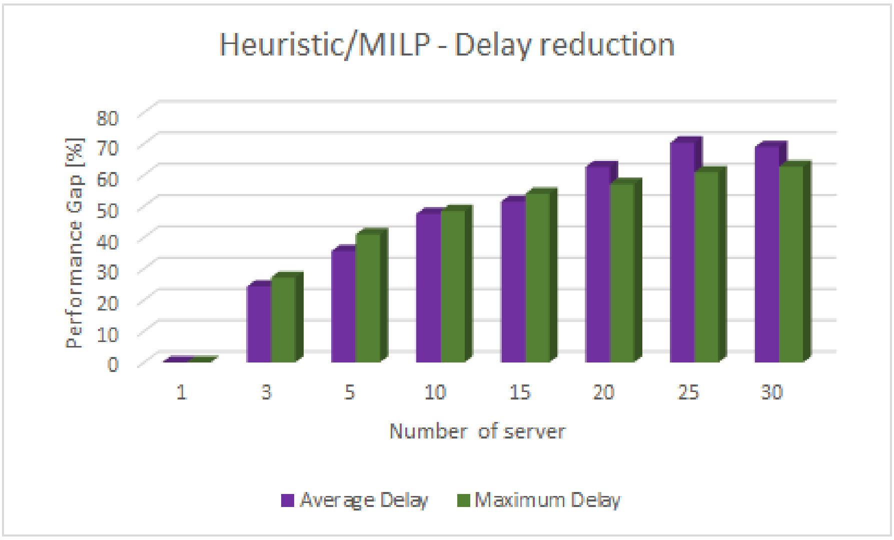


Figure 5.3: Varying the number of server nodes: performance gap - delay reduction.

- The performance gap analysis (see Fig. 5.3) quantifies the superior efficiency of the MILP1 approach in optimizing VM placement and minimizing network latency. For instance, as the number of servers increases, the MILP1 consistently outperforms the heuristic by substantial margins.
- In terms of *average delay reduction*, the MILP1 achieves a performance improvement starting from 24,3% with three servers, steadily increasing to a remarkable 68,9% reduction with 30 servers.
- Similarly, for the *maximum delay*, MILP1 demonstrates even greater reductions, starting at 27,2% for three servers and peaking at 62,7% with 30 servers. This consistent reduction highlights the robustness and scalability of the MILP1 approach, which becomes increasingly effective as the network topology scales.

4. **Insights and Implications**

- *MILP1 Offers Superior Delay Optimization*: Across both average and maximum delay metrics, MILP1 consistently delivers lower delay values, reflecting its optimized allocation and VM placement capabilities. MILP1, designed to find the global optimum for end-to-end latency minimization, performs better across all scenarios. Its mathematical formulation enables it to evaluate all possibilities and consistently find the best configuration, explaining its lower delays and reduced variability.
- *Heuristic Approach is Less Consistent*: While it achieves reasonable results, the heuristic method's wider ranges suggest a higher susceptibility to variability across simulation runs.
- *Critical Role of Maximum Delay*: The maximum delay metric is particularly relevant in ensuring system reliability, as it identifies worst-case scenarios that could disrupt synchronization in NMP systems.

5. **Final Remarks**

    - These results emphasize the importance of selecting the appropriate VM placement strategy to meet the stringent latency requirements of NMP systems. The MILP1 approach, while computationally more demanding, proves to be the superior choice for delay-critical applications, offering consistent and robust performance. On the other hand, the heuristic method provides a faster, even if less precise, alternative, making it suitable for scenarios where computational resources or time constraints are significant.
    - By presenting the average and maximum delay trends together, the analysis highlights both the **general improvements achieved by increasing server nodes** and the specific advantages of MILP1 in minimizing delay extremes. The insights gained from these results provide a strong foundation for further optimization and practical implementation of NMP systems in real-world scenarios.

5.1.2 Analysis and Explanation of Results: Simulator vs MILP2 (Maximum Server Load)

The provided figure (see Fig. 5.4) compares the Maximum Bandwidth Capacity in *Mbps* obtained through two different approaches (Heuristics and MILP) as the number of servers in a network topology increases. The data shows the averages obtained over 10 simulation runs, while the error bars represent the range of variation across these runs, highlighting the minimum and maximum values that each metric achieved during the simulations. The Maximum Server Load represents the highest bit rate handled by a single server node among all the server nodes in the network topology at any point during the entire simulation, reflecting the worst-case scenario for server load distribution.

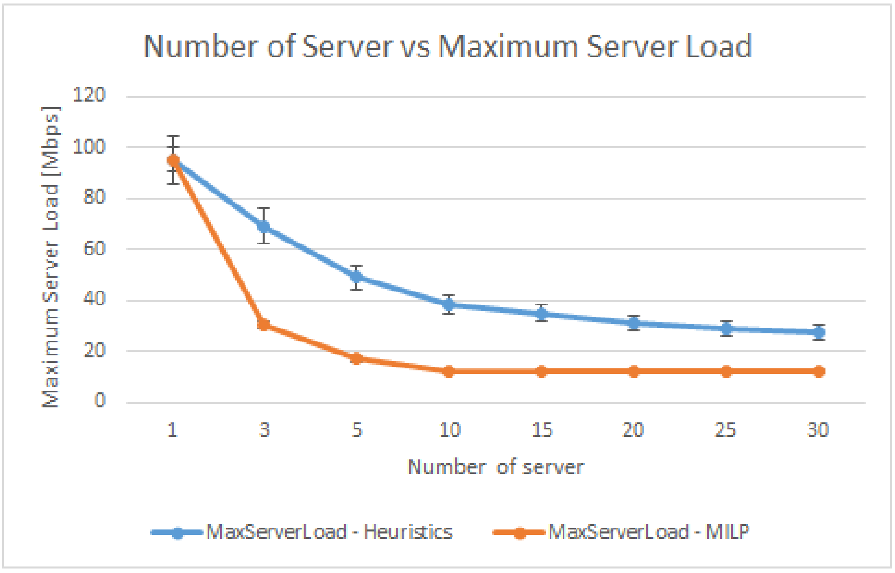


Figure 5.4: Varying the number of server nodes: maximum server load comparison.

Observations

1. *Heuristics*:

   - The Maximum Server Load starts at 95,26 Mbps for a single server and steadily decreases as the number of servers increases.
   - Despite the reduction in load, the maximum server load remains higher compared to the MILP approach across all scenarios.
   - The load decreases non-linearly, showing sharp reductions when the server count is small (e.g., from 1 to 5 servers) because, with a small number of servers, each server is required to handle a significant portion of the total traffic: all 30 musical events involving 50 musicians are distributed among a limited number of servers and this creates a bottleneck, especially if the maximum server load capacity is limited (in our experiment each server node can manage until 50 Gbps). However, beyond 10 servers, the reduction becomes more gradual.

2. *MILP2*:

   - The Maximum Server Load using MILP also starts at 95,26 Mbps for one server and decreases rapidly to stabilize at 12,29 Mbps after approximately five servers.
   - The MILP results demonstrate remarkable stability across all scenarios beyond 10 servers, indicating an efficient and optimal distribution of network load.
   - Error bars for MILP are negligible, suggesting consistent performance across all simulation runs.

3. *Comparison between Heuristics and MILP*:

- The heuristic simulator approach results substantially higher server loads compared to MILP across all configurations. For example, with 5 servers, the simulator loads is 49 Mbps, nearly 3 times higher than the MILP load (16,90 Mbps).
- The gap between the two methods narrows as the number of servers increases, but even with 30 servers, the heuristic approach shows a maximum server load (27,5 Mbps) that is still more than double higher than that of MILP (12,29 Mbps).
- MILP exhibits better load balancing, which minimizes the burden on any individual server, while the heuristic approach appears less effective in achieving load uniformity.
- The percentage reduction in Maximum Server Load achieved by MILP2 relative to the heuristic (see Fig. 5.5) demonstrates a substantial improvement. Starting from no difference when only one server is present, the MILP2 achieves a significant reduction of 56,6% with three servers, and the reduction peaks at 68% with ten servers. As the number of servers increases further, the reduction stabilizes, maintaining values around 55% to 65%, indicating a consistently superior performance of MILP2 in managing network traffic even in larger topologies.

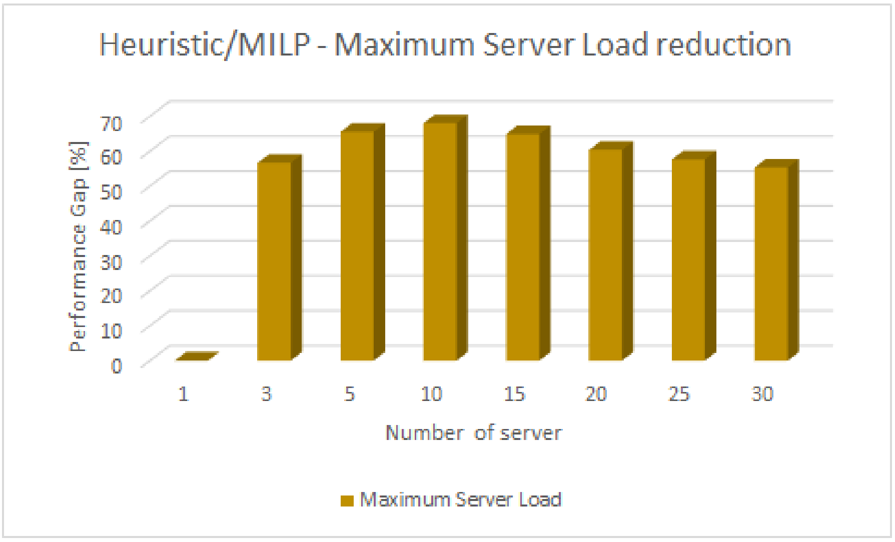


Figure 5.5: Varying the number of server nodes: performance gap - server load reduction.

Analysis:

- The results clearly demonstrate the superiority of the MILP approach in terms of optimizing the maximum server load across all configurations (excluding the case when there is only one server). This is primarily due to the ability of MILP to globally solve the load distribution problem by considering all constraints and network topology characteristics. The heuristic approach, while computationally faster and simpler, provides suboptimal solutions that fail to evenly distribute the load, particularly when the number of servers is low. As the network grows in terms of servers, simulator improves its performance but still lags behind MILP.

- Additionally, the range of values for heuristic simulator are larger compared to MILP, indicating higher variability in outcomes across the 10 simulation runs. This inconsistency could stem from the approximations used in simulator algorithms. In contrast, MILP's deterministic nature ensures consistent and predictable results with minimal variability even if the input random parameters (musical event duration and musicians involved per musical event) are the same for simulator and MILP model.

Implications:

1. *Scalability*:

    - MILP achieves near-optimal load balancing even in networks with a small number of servers, ensuring no single server is excessively burdened.
    - Heuristic, though less optimal, may be more practical for large-scale networks where computational efficiency is prioritized over absolute optimization.

2. *Applicability*:

    - For scenarios demanding high reliability and low latency, such as critical network infrastructures, MILP is the preferred approach due to its precise load balancing and stability.
    - Heuristic can be a viable option for less critical applications, when computational resources are limited.

Conclusion: The MILP approach outperforms heuristic simulator in achieving balanced server loads, especially in smaller networks. However, as the network grows, simulator demonstrates gradual improvement but still falls short of MILP's performance.

5.1.3 Analysis and Explanation of Results: Simulator vs MILP1 vs MILP2 (Execution Time)

The provided graph (see Fig. 5.6) compares the execution times of three approaches: Heuristics, MILP1, and MILP2, in terms of execution time (in seconds) as the number of servers in the network topology increases.

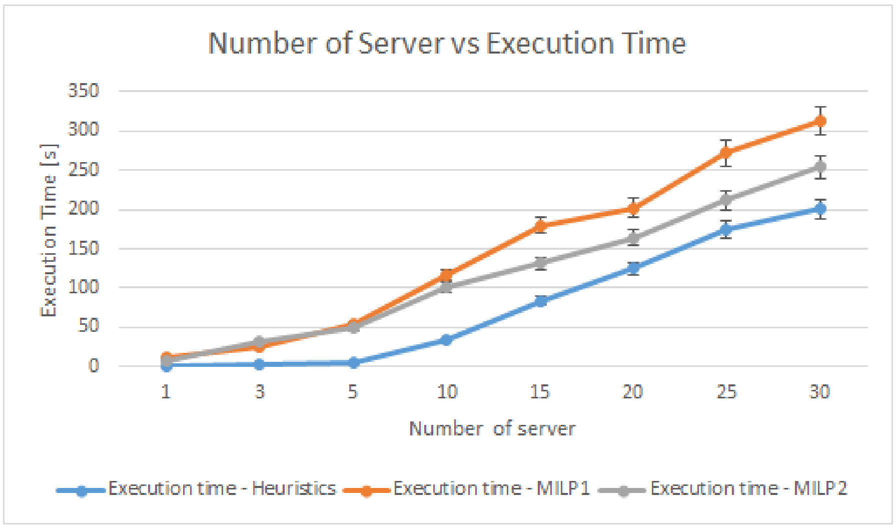


Figure 5.6: Varying the number of server nodes: execution time comparison.

Observation and Trends

1. *Heuristic Approach*:

    - The heuristic approach consistently exhibits the lowest execution times across all scenarios, demonstrating its computational efficiency.
    - The execution time increases approximately linearly with the number of servers, with a maximum execution time of 201 seconds for 30 servers.
    - This result is expected, as heuristic methods typically prioritize speed over precision, avoiding computationally intensive optimizations.

2. *MILP1*:

    - MILP1 has significantly higher execution times than the heuristic approach, reflecting the higher computational complexity associated with solving exact optimization problems.
    - The execution time grows non-linearly with the number of servers, reaching 313 seconds for 30 servers.
    - The increasing gap between MILP1 and the heuristic approach as the number of servers increases highlights the scalability challenge of MILP1.

3. *MILP2*:

    - MILP2 shows an intermediate behavior, with execution times consistently lower than MILP1 but higher than the heuristic method.
    - Its execution time also grows non-linearly, reaching 254,2 seconds for 30 servers, indicating it is more computationally efficient than MILP1.

4. *Performance Gap Analysis*:

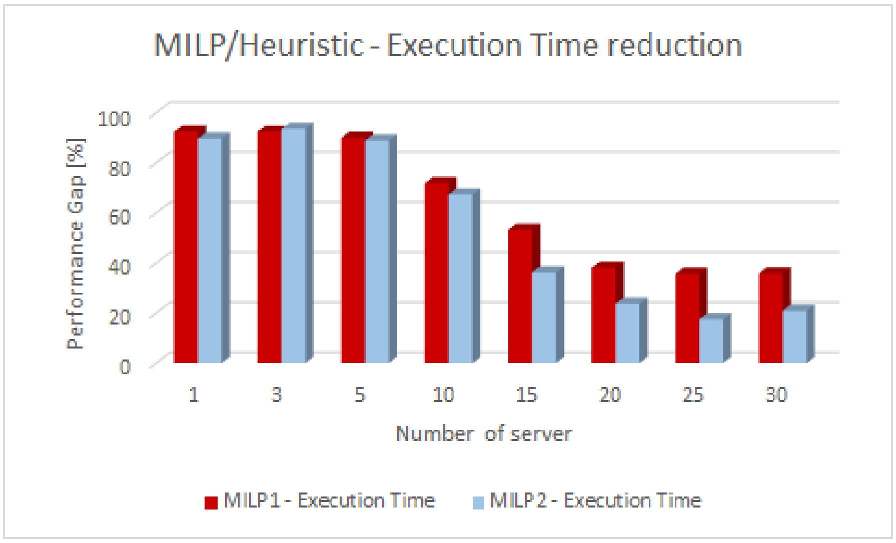


Figure 5.7: Varying the number of server nodes: performance gap - execution time reduction.

- The percentage reduction in execution time achieved by the heuristic compared to MILP1 (see Fig. 5.7) starts remarkably high, at 92,4% with one server, and remains consistent above 89% for three and five servers. As the number of servers increases, the gap narrows, with reductions of 71,8% for ten servers, 53,3% for fifteen servers, and 35,7% for thirty servers, indicating that MILP1 becomes relatively more computationally efficient as the topology grows larger.
- Similarly, the heuristic also shows significant execution time reductions when compared to MILP2 (see Fig. 5.7). Starting with 89,6% for one server, the reduction stabilizes around 88% to 93% for up to five servers, demonstrating high efficiency for small-scale topologies. As the number of servers increases, the reduction decreases, reaching 67,3% for ten servers, 36,3% for fifteen servers, and finally 20,9% for thirty servers. This trend reflects that MILP2, though computationally less demanding than MILP1, still cannot match the heuristic's execution time for large topologies.

5. *Execution Time Variance*:

- The graph includes error bars representing the distribution of execution times over 10 simulation runs (minimum and maximum values).
- Variability in execution times is relatively small for all methods.

Explanation of Results

1. *Computational Complexity*:

  - The heuristic method uses simplified decision-making rules, avoiding the exhaustive search and mathematical rigor required by MILP methods, which explains its faster execution time.
  - MILP1 and MILP2 rely on mathematical models and optimization solvers, leading to higher computational demands. MILP2 appears to be a more refined or tailored version of MILP1, likely incorporating optimizations or simplifications that reduce execution time.

2. *Scalability*:

  - The heuristic approach scales almost linearly with the number of servers, making it suitable for large-scale problems where execution speed is a priority.
  - Both MILP1 and MILP2 exhibit less linear scalability, suggesting that their computational requirements grow rapidly as the network topology becomes more complex.

3. *Trade-off*:

  - The heuristic approach is ideal for scenarios where execution time is a critical factor, and precision can be compromised.
  - MILP1 and MILP2 are better suited for situations where achieving near-optimal solutions is critical, but they may become impractical for very large networks due to their execution time.

Conclusion: The **heuristic approach is the most efficient in terms of execution time**, making it a practical choice for large-scale networks. The decision on which method to use depends on the application requirements and the relative importance of execution time versus the specific optimization goals. The delay performance of the heuristic approach remains reasonably close to that of the MILP, making it a viable option in cases where minimizing server load is less critical than maintaining low delays. This **balance between execution time and approximation quality** reinforces the notion that the decision on which **method to use depends on the application requirements** and the relative importance of execution time versus specific optimization objectives.

5.2 Varying the Number of Musical Events

This scenario involves progressively increasing the number of simulated musical events while maintaining all other parameters unchanged, included the network topology. The goal of this experiment is to evaluate how the impact of additional traffic, in terms of musical events, influences the performance expressed such as *average delay*, *maximum delay*, *maximum server load* and *execution time.*

The parameters that remain constant in this scenario are shown in Tab. 5.2:

Number of musicians	50
Number of server nodes	15
Time horizon [min]	1000
Maximum number of participants per musical event	8
User bit rate [Mbps]	1,536
Maximum musical event duration [min]	120
Maximum bandwidth capacity per server node [Gbps]	50
Queueing and Processing delay per node [ms]	3
Buffer delay [ms]	5
Delay variation [%]	30
Maximum tolerable user-server latency [ms]	1000
Propagation speed in the optical fiber [Km/s]	199862

Table 5.2: Scenario 2: constant parameters meanwhile varying the number of musical events

5.2.1 Analysis and Explanation of Results: Simulator vs MILP1 (Delay)

The provided plots (see Fig. 5.8, 5.9) illustrate the trends in Average Delay and Maximum Delay as a function of the number of musical events within the network. Both metrics are measured for the heuristic approach and the MILP1 model, showing how the system performs under increasing traffic conditions. Each data point represents the average of 10 simulation runs, while the error bars indicate the range of values observed across these runs, specifically the minimum and maximum values, highlighting the variability of the simulation process.

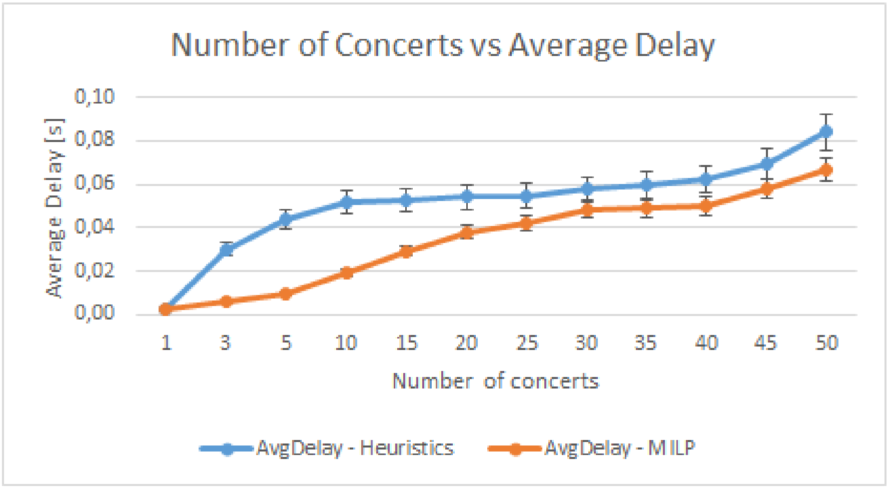


Figure 5.8: Varying the number of musical events: average delay comparison.

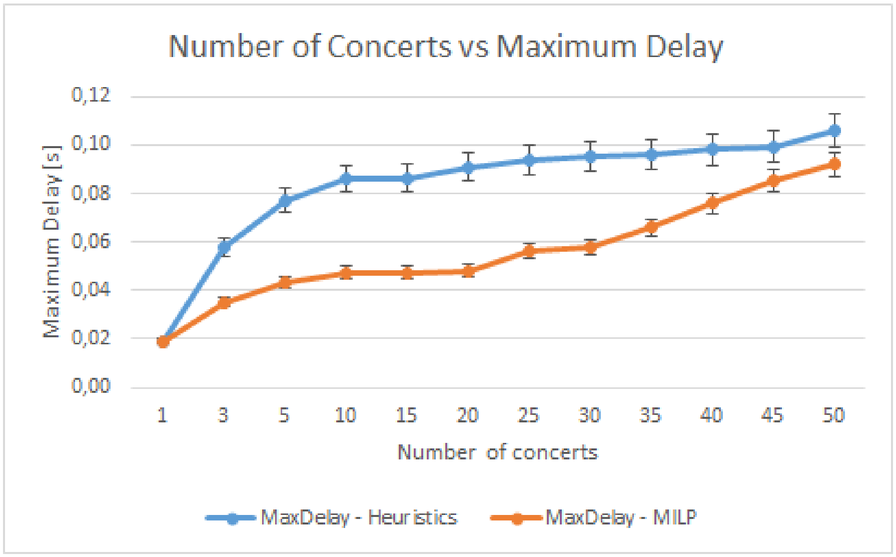


Figure 5.9: Varying the number of musical events: maximum delay comparison.

General Trends and Observations

1. *Average Delay*:

   - The Average Delay for both approaches increases as the number of concerts grows, which is expected because higher traffic levels introduce more challenges in managing end-to-end latency.
   - The heuristic approach, while simpler and less computationally expensive, lacks the precision of MILP, leading to higher average delays and greater variability. For example, at 50 concerts, the average delay reaches approximately 0,084 seconds for the heuristic approach but only 0,067 seconds for MILP1.
   - MILP1 achieves superior results because it optimizes the placement of VMs to minimize latency, ensuring a better allocation of network resources even under high traffic. The MILP approach scales more gracefully, maintaining lower delays and smaller growth compared to the simulator. This indicates that the MILP approach is better suited for managing higher loads in the network.

2. *Maximum Delay*:

   - Similarly, the Maximum Delay follows an upward trend as the number of concerts increases, reflecting the increased stress on the network and server nodes.
   - As observed with the Average Delay, the MILP1 model outperforms the heuristic approach. At 50 concerts, the maximum delay is 0,106 seconds for the heuristic approach, compared to 0,092 seconds for MILP1.
   - As the number of concerts grows, the maximum delay increases for both methods, but the growth rate differs:

  - The simulator exhibits a steep increase in maximum delay, starting at 0,058 seconds for 3 concerts and reaching 0,106 seconds for 50 concerts.
  - In contrast, the MILP approach grows more gradually, from 0,035 seconds for 3 concerts to 0,092 seconds for 50 concerts, highlighting its better scalability and ability to manage high loads effectively.

3. *Differences and Analogies*:

   - *Differences*:
     - While the trends for both Average and Maximum Delay are similar, the absolute values for Maximum Delay are always greater than the corresponding Average Delay values, as expected. This reflects the fact that Maximum Delay accounts for the worst-case scenarios across all nodes, while Average Delay represents the mean performance across the system.
     - The error bars for the heuristic approach are consistently larger for both metrics, indicating greater variability in the performance of the heuristic model compared to MILP1. This variability underscores the heuristic's suboptimal placement decisions, which lead to inconsistent performance across simulation runs.
   - *Analogies*:
     - In both metrics, the MILP1 model demonstrates better scalability and robustness to increased network traffic. Its ability to minimize both average and worst-case delays makes it a more suitable choice for applications with stringent latency requirements, such as NMP systems.

4. *Performance Gap Analysis*:

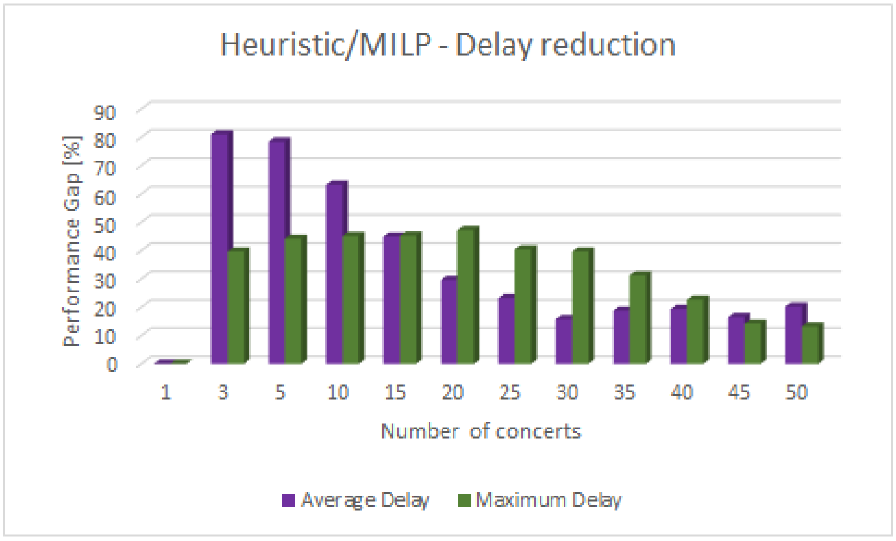


Figure 5.10: Varying the number of of musical events: performance gap - delay reduction.

- In terms of average delay (see Fig. 5.10), the reductions are most significant for low numbers of concerts, with 81% reduction at three concerts and 78,4% at five concerts. As the number of concerts increases, the percentage reduction gradually decreases, with values of 63,3% at ten concerts, 44,9% at fifteen concerts, and 29,6% at twenty concerts. Beyond this point, the reductions become less pronounced, stabilizing at around 15,8% to 20,2% for thirty to fifty concerts. This trend indicates that the MILP1 model's advantage diminishes as network traffic intensifies, likely due to increased congestion impacting both approaches.
- For maximum delay (see Fig. 5.10), the reductions are also noticeable but less dramatic compared to average delay. For three concerts, MILP1 achieves a 39,7% reduction, increasing to 44,2% at five concerts and 45,1% at ten concerts. Similar to average delay, the improvement stabilizes for higher traffic levels, with reductions of 31,3% at thirty-five concerts, 22,7% at forty concerts, and 13,2% at fifty concerts. This trend suggests that while MILP1 maintains a performance edge over the heuristic approach, the advantage narrows as the number of concerts and the resulting traffic load increase.

Importance of the Results:

- The analysis confirms that the **MILP1 model outperforms the heuristic approach in handling the increasing complexity of network traffic as the number of concerts grows**. This finding aligns with the stringent requirements of NMP systems, where delays must be kept under 25–30 milliseconds to maintain musical synchronization. While the heuristic approach can offer faster execution times, its inferior performance in managing delay highlights its limitations in scenarios with high traffic density.

- The heuristic approach, while computationally less intensive, relies on approximate solutions that do not guarantee optimality. As a result, it is more prone to resource contention and suboptimal scheduling decisions, leading to higher maximum delays, especially when the network is under heavy load. In contrast, The MILP approach outperforms the heuristic method because it uses a globally optimized allocation of resources and scheduling. This **minimizes bottlenecks in the network, which are particularly critical in high-load scenarios** with multiple simultaneous concerts. Its ability to manage the network more efficiently explains the slower growth in maximum delay and its consistent performance, as shown by the smaller error bars.

- In conclusion, these results demonstrate that **MILP1 offers a significant advantage in managing network resources under high traffic conditions**, reducing both the average and maximum delays and maintaining a more consistent performance profile across multiple simulation runs.

5.2.2 Analysis and Explanation of Results: Simulator vs MILP2 (Maximum Server Load)

The graph (see Fig. 5.11) illustrates the Maximum Bandwidth Capacity (measured in *Mbps*) as the number of concerts increases in the network topology, comparing the performance of the heuristic simulator approach and the MILP model. The data reflects the averages calculated from 10 simulation runs, with error bars illustrating the variability observed across these runs, specifically showing the minimum and maximum values that each metric reached during the simulations.

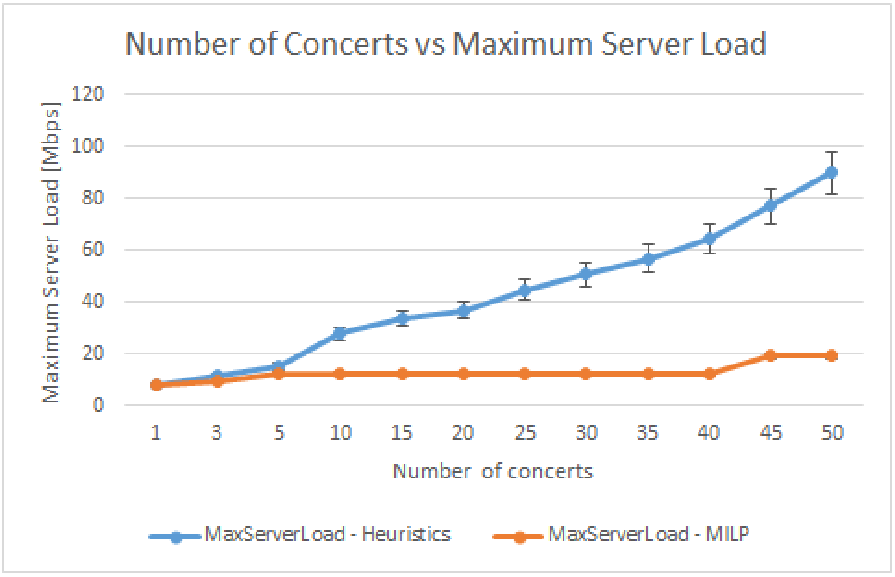


Figure 5.11: Varying the number of musical events: maximum server load comparison.

Observations

1. *Initial Performance*:
   - For a single musical event:
     - The heuristic approach and the MILP model yield comparable results (7,68 Mbps).
     - Both approaches demonstrate low server load at this scale, suggesting they handle a minimal load effectively.
2. *Scalability*:
   - As the number of concerts increases, the heuristic approach results in a steep rise in maximum server load: the server load jumps to 34 Mbps for 15 concerts and escalates dramatically to 90 Mbps for 50 concerts.
   - In contrast, the MILP model maintains a nearly constant server load, staying close to 12 Mbps up to 40 concerts, and only showing a slight increase to 19,29

Mbps by 45 concerts. This disparity highlights the superior scalability of the MILP model.

3. *Performance Gap Analysis*:

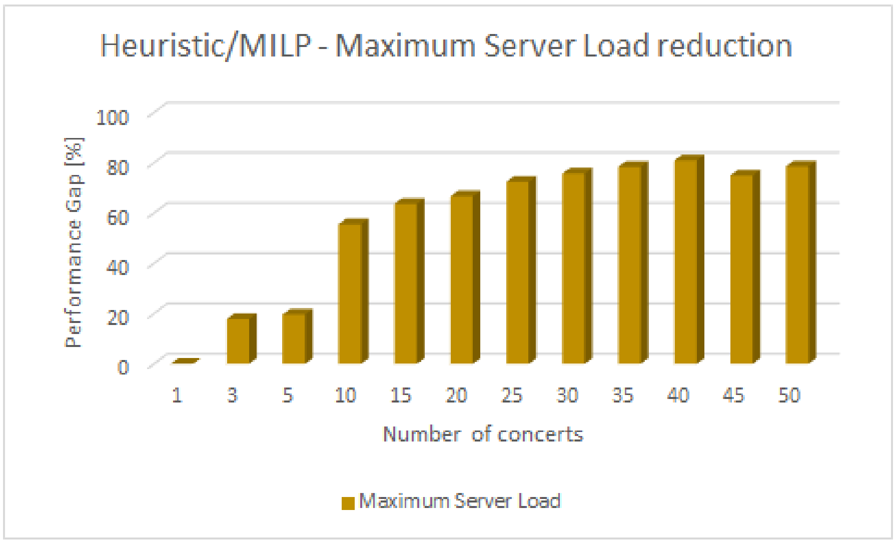


Figure 5.12: Varying the number of musical events: performance gap - server load reduction.

- The data demonstrates that MILP2 provides substantial reductions in Maximum Bandwidth Capacity compared to the heuristic approach, with the magnitude of these reductions increasing with higher numbers of concerts.
- For scenarios with a small number of concerts (see Fig. 5.12), the reductions are relatively modest. At three concerts, the reduction is 17,8%, and at five concerts, it reaches 19,6%. However, as the number of concerts grows, the reductions become more pronounced. At ten concerts, MILP2 achieves a 55,6% reduction, increasing to 63,6% at fifteen concerts, and peaking at 66,7% at twenty concerts.
- For even higher numbers of concerts (see Fig. 5.12), MILP2 consistently demonstrates a significant reduction in Maximum Server Load. The reductions reach 72,4% at twenty-five concerts, 75,8% at thirty concerts, and an impressive 78,4% at thirty-five concerts. The trend continues with a 78,6% reduction at fifty concerts, indicating the strong efficiency of MILP2 in high-traffic scenarios.

4. *Error Bars*:

- The error bars represent variability over 10 simulation runs:
  - The heuristic approach shows increasing variability as the number of musical events grows, particularly at higher loads, indicating less predictable performance.

  - The MILP model demonstrates minimal variability, with tightly clustered results even at high loads, reflecting its robust and consistent performance.

Explanation

1. *MILP Superiority*:
   - The MILP model achieves significantly lower maximum server loads due to its ability to optimize resource allocation across the network. By efficiently distributing the load among servers, it prevents overloading any single server, even as the number of concerts increases.
   - This explains the model's nearly constant server load up to 40 concerts and its slow, controlled increase thereafter.
2. *Heuristics Limitations*:
   - The heuristic approach, by contrast, lacks global optimization and tends to assign resources in a less balanced way. As the network becomes more congested with an increasing number of concerts, this results in certain servers becoming heavily overloaded.
   - This accounts for the steep and uncontrolled rise in maximum server load observed for the simulator.
3. *Impact of Network Load*:
   - The heuristic approach struggles to scale effectively as the number of concerts increases, leading to severe bottlenecks and overburdening of specific servers (this is critical for servers that have limited maximum server load capacity).
   - The MILP model, on the other hand, demonstrates excellent scalability, effectively managing the increased network load while keeping the maximum server load well below critical levels (in our experiment each server node can manage until 50 Gbps but in a real scenario this value can be much lower).

Conclusion: The comparison highlights the clear advantage of the MILP model in managing server load under increasing numbers of musical events. While the heuristic approach may suffice for smaller networks with limited concerts, it quickly becomes infeasible as the number of events grows, leading to severe server overloading. In contrast, the MILP model provides a more sustainable and scalable solution, ensuring balanced resource allocation and maintaining low server loads even under high-demand scenarios.

5.2.3 Analysis and Explanation of Results: Simulator vs MILP1 vs MILP2 (Execution Time)

The plot (see Fig. 5.13) illustrates the Execution Time (in seconds) for three approaches, heuristic simulator, MILP1, and MILP2, as the number of concerts increases in the network topology.

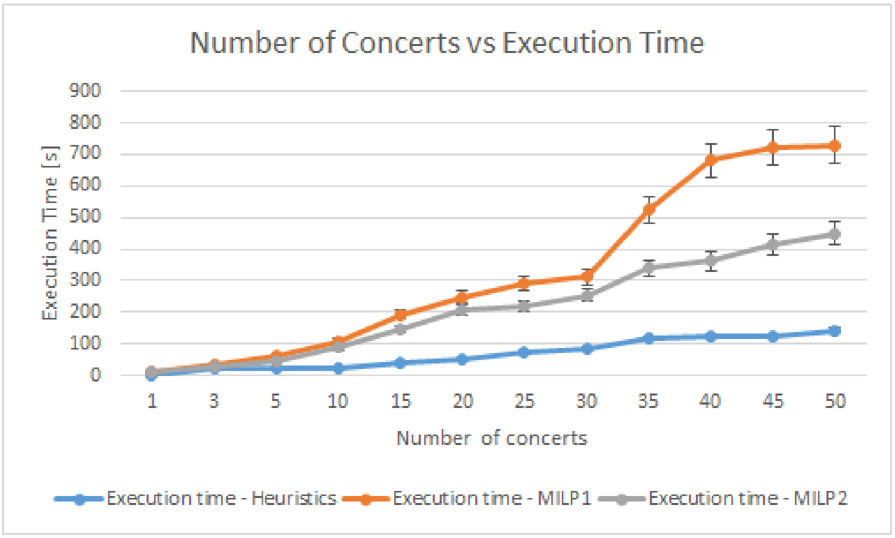


Figure 5.13: Varying the number of musical events: execution time comparison.

Observations

1. *Initial Execution Times*:
   - For a single concert, heuristic approach executes in 0,263 seconds, which is significantly faster compared to MILP1 (10,7 seconds) and MILP2 (8,6 seconds).
2. *Scalability*:
   - As the number of concerts increases:
     - Simulator demonstrates minimal growth in execution time, increasing almost linearly from 0,263 seconds (1 concert) to 142 seconds (50 concerts).
     - MILP1 experiences a sharp rise, particularly beyond 30 concerts, reaching 730 seconds at 50 concerts.
     - MILP2, though faster than MILP1, also shows significant growth, peaking at 449 seconds for 50 concerts. However, its growth is less steep than MILP1.
3. *Performance Gap Analysis*:
   - For a single concert (see Fig. 5.14), the heuristic approach is significantly faster, with execution times 97,5% lower than MILP1 and 96,9% lower than MILP2. This trend reflects the simplicity and speed of the heuristic, which requires less computational overhead. As the number of concerts increases, the percentage advantage decreases, although the heuristic continues to outperform the MILP models. For three concerts, the heuristic's execution time remains 42,9% lower than MILP1 and 31% lower than MILP2.
   - As network traffic grows (see Fig. 5.14), the heuristic retains a consistent edge. At five concerts, its execution time is 61% faster than MILP1 and 45,2% faster

than MILP2. For ten concerts, the advantage increases further, with reductions of 77,8% compared to MILP1 and 72,4% compared to MILP2. These values indicate the scalability of the heuristic method in scenarios with moderate traffic.

- For fifteen and twenty concerts (see Fig. 5.14), the heuristic remains significantly faster, with reductions of 80,7% and 80,2% compared to MILP1, and 74,1% and 76,2% compared to MILP2, respectively. Even as the number of concerts reaches higher levels, such as thirty or fifty, the heuristic maintains a substantial execution time advantage. At thirty concerts, it is 73,1% faster than MILP1 and 66,9% faster than MILP2, while at fifty concerts, the reductions remain 80,5% and 68,4%, respectively.
- These trends reflect that MILP2, though computationally less demanding than MILP1, still cannot match the heuristic's execution time for large network traffic.

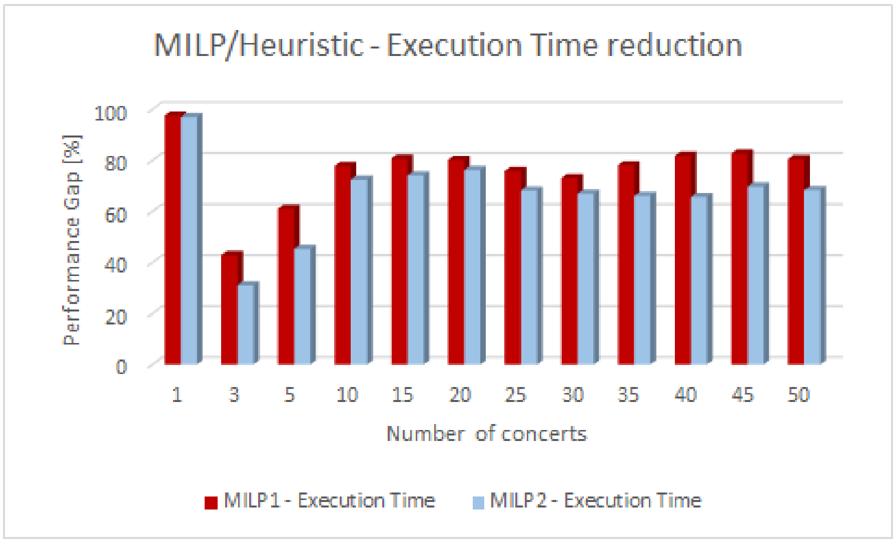


Figure 5.14: Varying the number of musical events: performance gap - execution time reduction.

Explanation

1. *Heuristics Efficiency*:

   - The heuristic approach uses a less computationally intensive algorithm, which explains its rapid execution and near-linear growth in execution time. However, this comes at the cost of lower solution accuracy and suboptimal resource allocation (as observed in other metrics like delay and server load).

2. *MILP1 and MILP2 Complexity*:

   - Both MILP1 and MILP2 use mathematical optimization techniques that require solving complex equations to ensure optimal solutions:

3. *Scalability Challenges*:
    - For large numbers of concerts (e.g., 35–50), the execution time of both MILP1 and MILP2 grows rapidly due to the increased computational burden of handling a larger optimization problem.
    - In contrast, the heuristic approach continues to scale near-linearly, making it more suitable for time-sensitive applications at the cost of solution quality.
4. *Error Bars*:
    - The error bars for MILP1 and MILP2 (see Fig. 5.13) indicate variability, especially at higher loads, whereas the simulator approach maintains a consistent execution time with narrow error bars. Each error bar shows the minimum and maximum values that each metric reached during the simulations. The wider error bars for MILP1 and MILP2 at larger numbers of concerts reflect variability in computational effort due to differences in problem complexity.

Conclusion: For small-scale systems (e.g., <10 concerts), all methods are viable, though simulator is preferable for quick results; for large-scale systems (e.g., >30 concerts), the heuristic approach may be preferable in time-sensitive scenarios, while MILP1 and MILP2 are better suited for applications prioritizing solution quality. This analysis underscores the importance of selecting the appropriate method based on the network's size and the specific requirements for accuracy versus execution time.

5.3 Final Comments

The analysis of NMP systems has highlighted the critical importance of minimizing end-to-end delay. Specifically, when the delay exceeds 25–30 ms, musical synchronization is disrupted, significantly impairing the performance quality. This underscores the need for careful selection of the number of musical events that can be processed within a specified time horizon, given a particular network topology.

The comparison between MILP models and simulation-based approaches reveals distinct advantages of the former in optimizing latency and server load. **MILP2 model**, in particular, consistently outperforms the simulator in **reducing the maximum server load**, which is a critical metric for maintaining system scalability and efficiency. It exhibits a remarkable capability to anticipate and allocate server resources efficiently, resulting in maximum server loads that are up to **5 times lower** (see Fig. 5.11) than those observed with the simulator. This advantage is particularly pronounced even in demanding scenarios, such as managing 50 concerts over 1000 minutes with 50 musicians and 15 server nodes. Similarly, in terms of latency, **MILP1** demonstrates superior performance, achieving **latency reductions of up to 78%** (see Fig. 5.10) compared to the simulator.

However, the **computational complexity** of MILP models presents a notable trade-off. Their slower execution times can be a limitation in real-time applications, especially for large-scale or complex network topologies. In such cases, the simulator offers a viable alternative, delivering suboptimal solutions within a reasonable time frame. This trade-off between solution optimality and computational efficiency highlights the importance of tailoring the approach to the specific constraints and requirements of the NMP system being developed.

Moreover, the **simulator serves as a powerful tool for analyzing the feasibility of real-world scenarios** by adjusting input parameters. It allows for testing various configurations to ensure that both latency constraints (staying below 25-30 ms to maintain functional NMP scenarios) and server load constraints (since nodes often have limited capacity for simultaneous Bit Rate management) are met before actual implementation. This ability to simulate and evaluate potential configurations provides valuable insights into system behavior, enabling the identification of bottlenecks and limitations in a controlled, risk-free environment.

Furthermore, the data and plots illustrate additional insights into the system's behavior under varying conditions. For instance, the relationship between network topology, musical event density, and performance metrics suggests that **strategic adjustments in network configuration can further enhance performance**. These findings emphasize the need for a holistic approach that integrates both model-based optimization and practical simulations to address the multifaceted challenges of NMP systems, ultimately bridging the gap between theoretical modeling and real-world implementation.

Chapter 6

Conclusions

This thesis focused on **evaluating the performance of NMPs over a 5G-enabled optical network infrastructure**. By addressing the critical challenges of latency minimization and server load balancing, it aimed to ensure the feasibility of high-quality, synchronous musical interactions among geographically distributed participants. The core of this work was the development of a **Python-based heuristic simulator and two MILP models**, leveraging **Gurobi** as optimization library, to analyze and optimize system performance.

The importance of this research lies in the stringent requirements of NMP systems, where delays exceeding 25–30 ms disrupt musical synchronization, undermining the performance experience. To tackle this, the work began by **exploring the state-of-the-art**, which emphasized the role of 5G technologies such as URLLC, MEC, and network slicing as enablers for NMP applications. These features provide low-latency and high-reliability communication, essential for real-time audio streaming. To support these characteristics, we have provided detailed numerical results [15] [7] that underscore the acceptable latency threshold for NMP applications. These results demonstrate the crucial need to stay below the 25–30 ms delay limit to ensure the feasibility and quality of networked musical interactions. Complementing this, the **theoretical background delved into the technical foundations of 5G RAN and optical networks**. The combination of these technologies ensures scalability, reliability, and minimal delay in data transmission, making them ideal for latency-sensitive use cases like NMP.

At the heart of this thesis was the development of a **simulation framework and optimization models**. The heuristic simulator dynamically placed VMs serving as audio mixing servers based on real-time latency and load conditions. It prioritized low-latency paths, ensuring efficient resource use while maintaining simplicity and speed. On the other hand, the MILP models offered optimal solutions by leveraging global information about the network and future demand. MILP1 focused on minimizing the maximum end-to-end latency, while MILP2 aimed to minimize the maximum bandwidth capacity per server node, addressing two distinct but complementary objectives.

The experiments revealed significant insights into the performance trade-offs between the heuristic simulator and MILP models. **The MILP approaches demonstrated**

their superiority in optimizing both latency and server load. MILP1 reduced latency of up to 78% compared to the simulator, achieving nearly optimal synchronization among participants. Similarly, MILP2 exhibited exceptional load-balancing capabilities, particularly in high-demand scenarios, such as managing 50 concerts involving 50 musicians over 1000 minutes. In such cases, the simulator's maximum bandwidth capacity was up to 5 times higher than MILP2's, highlighting the advantages of the latter's foresight in resource allocation.

However, the MILP models also came with their limitations, primarily their higher computational complexity and slower execution times. These factors make them less practical for real-time applications or large-scale networks where rapid decisions are required. In contrast, the heuristic simulator, while delivering suboptimal solutions, was far more computationally efficient, making it a viable choice for scenarios with tight time constraints or lower resource demands. This trade-off underscores the **need for selecting the appropriate approach based on specific application requirements**.

Beyond evaluating the current models, this thesis also provides a platform for future improvements. One promising direction is enhancing the simulator to allow dynamic VM relocation during ongoing concerts, adapting to network changes such as congestion or degradation. Introducing varying server load capacities would create a more realistic simulation environment, reflecting the heterogeneity of real-world networks. Another exciting avenue is the integration of **Machine Learning (ML) to predict network conditions and optimize VM placement more intelligently**. Hybrid approaches, combining the efficiency of heuristics with the precision of MILP models, could further enhance the system's adaptability and scalability.

Bibliography

[1] *Centralized Unit – Distributed Unit (CU-DU) Split Base Station.* https://www.techtrained.com.

[2] *Decoding OLT, ONU, ONT, and ODN in PON Network.* https://community.fs.com/article/abc-of-pon-understanding-olt-onu-ont-and-odn.html.

[3] *The Importance of Simulating PON Networks Before Deployment.* https://www.m2optics.com/blog/passive-optical-networks-pons-why-its-important-to-test-them-before-deployment.

[4] *What are ORAN, DRAN, cRAN, and vRAN?* https://stl.tech/blog/what-are-oran-dran-cran-and-vran/.

[5] DANIEL DIK; MICHAEL STÜBERT BERGER. *Open-RAN Fronthaul Transport Security Architecture and Implementation.* 2023.

[6] Chris Chafe; Juan-Pablo Caceres. *JACKTRIP: UNDER THE HOOD OF AN ENGINE FOR NETWORK AUDIO.* 2010.

[7] Luca Turchet; Paolo Casari. *Assessing a Private 5G SA and a Public 5G NSA Architecture for Networked Music Performances.* 2023.

[8] Luca Turchet; Paolo Casari. *On the Impact of 5G Slicing on an Internet of Musical Things System.* 2024.

[9] Fabienne Saliou; Philippe Chanclou; Luiz Anet Neto; Gael Simon; Jeremy Potet; Mathilde Gay; Laurent Bramerie; Helene Debregeas. *Optical access network interfaces for 5G and beyond.* 2021.

[10] PETAR POPOVSKI; KASPER FLØE TRILLINGSGAARD; OSVALDO SIMEONE; GIUSEPPE DURISI. *5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View.* 2018.

[11] Chris Chafe; Juan-Pablo Caceres; Michael Gurevich. *Effect of temporal separation on synchronization in rhythmic performance.* 2010.

[12] Xiang Liu. *Enabling Optical Network Technologies for 5G and Beyond.* 2022.

[13] Luca Turchet; Claudia Rinaldi; Carlo Centofanti; Luca Vignati; Cristina Rottondi. *5G-enabled Internet of Musical Things Architectures for Remote Immersive Musical Practices.* 2024.

[14] CRISTINA ROTTONDI; CHRIS CHAFE; CLAUDIO ALLOCCHIO; AUGUSTO SARTI. *An Overview on Networked Music Performance Technologies.* 2016.

[15] Luca Vignati; Giovanni Nardini; Marco Centenaro; Paolo Casari; Sandra Lagén; Biljana Bojovic; Stefano Zambon; Luca Turchet. *Is Music in the Air? Evaluating 4G and 5G Support for the Internet of Musical Things.* 2023.

[16] Jan Durre; Robert Hupke; Norbert Werner. *In-Depth Latency and Reliability Analysis of a Networked Music Performance over Public 5G Infrastructure.* 2022.

Acknowledgements

Ringrazio i miei genitori per avermi sostenuto in questo percorso. Senza di loro questo traguardo non sarebbe stato possibile. Gli sarò sempre grato per avermi lasciato libero di fare le mie scelte.

Ringrazio i miei fratelli, il mio porto sicuro, la mia costante, l'indelebile che ho sulla pelle. Grazie di esistere.

Ringrazio Alessandra, la mia compagna di vita, la persona a cui ho sottratto più tempo, per essermi sempre stata vicina nei momenti più difficili e per aver alleggerito questi anni con la sua voglia di viaggiare.

Ringrazio i miei amici per essere sempre rimasti al mio fianco nonostante la distanza. Ringrazio altresì gli amici che ho ritrovato al nord e le nuove amicizie, per avermi fatto sentire a casa.

Ringrazio i miei colleghi universitari, i miei compagni di viaggio, per aver condiviso con me le gioie e le delusioni.

Ringrazio la mia relatrice, la professoressa Cristina Rottondi, per la sua estrema disponibilità e per i consigli utili che mi ha elargito.

Infine, ringrazio me stesso per non aver mai mollato, neanche quando i motivi per farlo si facevano sempre più ingombranti e affollavano la mia testa di giorno e di notte. L'unico motivo che mi ha spinto ad andare avanti fino alla fine è stato il voler evitare al me stesso di domani questo rimpianto.