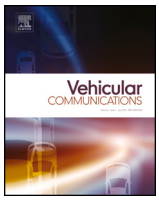
Vehicular Communications 45 (2024) 100691

Contents lists available at ScienceDirect

Vehicular Communications

journal homepage: www.elsevier.com/locate/vehcom



Network slice allocation for 5G V2X networks: A case study from framework to implementation and performance assessment

Ciprian Zamfirescu [a], Radu Iugulescu [a], Răzvan Crăciunescu [a,b,d], Alexandru Vulpe [a,b], Frank Y. Li [c,*], Simona Halunga [a]

[a] Department of Telecommunications, University Politehnica of Bucharest (UPB), 061071 Bucharest, Romania
[b] Beam Innovation SRL, 041386 Bucharest, Romania
[c] Dept. of Information and Communication Technology, University of Agder (UiA), N-4885 Grimstad, Norway
[d] Academy of Romanian Scientists, 050044 Bucharest, Romania

ARTICLE INFO

Keywords:
5G
Network slicing
V2X communications
Implementation and simulation
Performance assessment

ABSTRACT

Empowered by the capabilities provided by fifth generation (5G) mobile communication systems, vehicle-to-everything (V2X) communication is heading from concept to reality. Given the nature of high-mobility and high-density for vehicle transportation, how to satisfy the stringent and divergent requirements for V2X communications such as ultra-low latency and ultra-high reliable connectivity appears as an unprecedented challenging task for network operators. As an enabler to tackle this problem, network slicing provides a power tool for supporting V2X communications over 5G networks. In this paper, we propose a network resource allocation framework which deals with slice allocation considering the coexistence of V2X communications with multiple other types of services. The framework is implemented in Python and we evaluate the performance of our framework based on real-life network deployment datasets from a 5G operator. Through extensive simulations, we explore the benefits brought by network slicing in terms of achieved data rates for V2X, blocking probability, and handover ratio through different combinations of traffic types. We also reveal the importance of proper resource splitting for slicing among V2X and other types of services when network traffic load in an area of interest and quality of service of end users are taken into account.

1. Introduction

Unlike previous generations of mobile communication systems, the fifth generation (5G) systems introduce network virtualization and logical computing in order to facilitate emerging applications that may have diverse service requirements. As such, the traditional practice for medium sharing in a *one-size-fits-all* manner does not fit this type of networks. In the landscape of 5G technologies, network slicing is one of the key enablers for service provisioning and it represents an architectural trend for mobile network evolution [1][2]. Although mobile operators still provide all types of services based on a common physical network infrastructure, they are able to partition the entire network into multiple logical slices through network slicing, providing dedicated resources to customers to ensure their quality of service (QoS) [3].

Network slicing splits a physical network into multiple logical networks by creating multiple unique logical and virtualized networks over a common multi-domain infrastructure. Through this concept, a slice can be regarded as a set of programmable resources which is able to implement network functions and even acts like a complete end-to-end network, delivering specific services for its customers. Isolated from other slices based on the same infrastructure, the delivery of QoS requested by end users is accomplished in a slice through proper resource partitioning across slices as well as resource allocation inside the slice.

In the meantime, various vehicle-to-everything (V2X) applications ranging from autonomous driving, vehicle platooning, to broadband services are emerging. However, these V2X applications have diverse requirements. Coupled with difficult conditions such as high mobility and high density of vehicles, stringent and divergent requirements, e.g., low latency (under 10 ms), high reliability (near 100%), and high data rate (in the order of Gbps) may appeal [4][5]. Thanks to the flexible capability provided by 5G technologies, network slicing appears as a promising approach to fulfill the requirements for V2X services and applications [6][7].

* Corresponding author.
E-mail address: frank.li@uia.no (F.Y. Li).

https://doi.org/10.1016/j.vehcom.2023.100691
Received 2 March 2023; Received in revised form 27 July 2023; Accepted 18 November 2023
Available online 27 November 2023

In recent years, there has been a surge of activities in the research community targeting at network slicing based solutions for various applications including the automotive vertical for V2X applications [5][8]. Existing work on network slicing addresses various techniques to enable this concept, along with activities on standardization, architecture, and technical solutions, in addition to use cases. Although resource management for network slices has been investigated in various studies, there is so far not much work addressing the problem of optimizing slice allocation among a specific V2X slice and the rest of network slices. When heterogeneous traffic types, e.g., enhanced mobile broadband (eMBB), ultra-reliable low latency communication (URLLC), massive machine-type communication (mMTC)), and V2X traffic coexist, optimal slice resource allocation may become a problem. Such a problem becomes more interesting and challenging when a deployed real-life network is considered. This observation triggered our motivation for this study.

In this paper, we propose a framework for network slicing for V2X applications considering heterogeneous traffic types including V2X, eMBB, URLLC, and mMTC. Within this framework, several algorithms dedicated to clients (user equipment (UE)) and base stations (BSs) (next generation Node B (gNB) in the context of 5G. The terms of BS and gNB are interchangeably used in this paper) have been developed and implemented. Considering a real-life use case based on a deployed 5G network in an area of interest in the city of Bucharest, Romania, we investigate extensively the performance of the framework under various traffic conditions through simulations. Furthermore, we reveal which configurations are preferable for network slicing with respect to the defined performance parameters, by comparing the outputs of the algorithms with the service requirements.

Distinct from most existing work which focused on one aspect of network slicing, our work contains framework design, algorithm development, as well as implementation and validation through simulations based on real-life network scenarios. Briefly, the main contributions to this paper are summarized as follows:

- We propose a framework for network slicing with an emphasis on a dedicated V2X slice. The co-existence of other traffic types is considered in our framework.
- We develop a set of detailed algorithms for both base stations and UEs. The algorithms support flexible parameter configurations and are implemented in Python for both clients and base stations.
- Two scenarios based on real-life network deployment are identified and the performance of the framework is evaluated through extensive simulations.

Moreover in this work, the improvement brought by network slicing in 5G networks is studied. For performance assessment, three parameters for network slicing, i.e., data rate, handover ratio, and blocking probability, are considered. To obtain convincing numerical results, the positions of base stations in a real-life network and the movement of the clients are taken into consideration in our simulations.

The remainder of this paper is organized as follows. After summarizing related work in Section 2, we introduce the network slicing architecture and give background information on the types of services considered in this study in Section 3. Then Section 4 elaborates the proposed network slicing framework and algorithms. Afterwards, Sections 5 and 6 present and discuss the performed simulations and the obtained numerical results respectively. Based on these results, Section 7 explores a few aspects related to the applicability of the proposed framework. Finally, Section 8 draws conclusions and sheds light on future work.

2. Related work

In this section, we summarize related work which is split into two categories: architectures with various requirements, and network slicing based vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) solutions.

2.1. Architectures for SDN, NFV, MEC, and slicing

The work by [3] provides technical details in terms of updated solutions related to 5G network slicing using software defined networking (SDN) and network function virtualization (NFV) enablers, along with multi-access edge computing (MEC), and cloud/fog computing. In [9], the authors presented the capabilities brought by MEC in 5G network slicing by allocating network resources near to end users and performing computing at the edge of the network. In [10], the authors implemented an end-to-end (E2E) network slice orchestration and management platform and evaluated its performance from both system-centric and user-centric perspectives. Targeting at Industry 4.0 scenarios, [11] presented a novel 5G based network slicing framework for interconnecting different industrial sites up to the extreme edge with different network slices. Furthermore, the researchers from the same group proposed a novel E2E network slicing orchestration system and a dynamic auto-scaling algorithm, enabling autonomous adaptation of resources in an E2E network [12]. The performance of their solution was evaluated through system-level simulations. Furthermore, [13] proposed another dynamic auto-scaling algorithm considering the non-standalone architecture where both fourth generation (4G) and 5G network components co-exist and analyzed its performance via Markov modeling.

Aiming at automatic slice management for vertical applications considering VNF placement, resource assignment, and traffic routing, [14] developed a queuing-based model and applied it at the network orchestrator to optimally match verticals' requirements based on available network resources. Moreover, [15] offered a logical architecture solution for 5G network slicing and discussed the evolution of network architecture based on SDN and NFV. Based on the network slicing architecture, the authors therein revised handover procedures in mobility management and proposed a mobility management mechanism to offer flexible and agile customized services in network slicing based 5G systems.

2.2. V2V and V2I solutions

As ultra-low latency is a mandatory requirement for autonomous driving, road side units (RSUs) enabled by MEC are able to handle data coming from passing vehicles or platoons of vehicles that communicate with each other, not influencing the trajectory or the handling of the vehicles [16][17]. Focusing on V2X and V2I applications, [18] proposed a network slicing based communication model for vehicular networks including two logical slices, one autonomous driving slice and one infotainment slice, based on a common infrastructure. In [7], the authors presented an algorithm for resource allocation by taking into consideration mainly the communications between vehicles and infrastructure and between vehicles, with URLLC and eMBB slices.

In [19][20], the authors gave comprehensive surveys on various aspects for 5G V2X applications and vehicular networks. In [21], a lightweight blockchain inspired trust system was proposed for vehicle networks. Furthermore, [22] presented a network slicing system developed for 5G V2V networks that targeted to improve the performance of existing networks by involving unmanned aerial vehicles (UAVs) as aerial nodes. Their solution facilitates the communication between base stations and vehicular nodes, offering additional resources through UAVs. Therein, an optimization algorithm was developed for the positions of UAVs, and the performance evaluation had been made based on a simulator known as a simulator of urban mobility (SUMO). Targeting at autonomous driving for light-rails and tramways systems, the authors of [23] proposed a mechanism for inter- and intra-slice radio isolation for QoS guarantee. Therein, performance evaluation in terms of packet delivery ratio and required bandwidth with guaranteed delay was performed through a system-level 5G-air-simulator.

Table 1
Network slicing enabled solutions for V2X applications: A qualitative comparison

Solution versus Features	Our framework	[16]	[18]	[22]	[23]
Application scenarios	Slicing for eMBB, URLLC, mMTC, V2X	Cellular V2X for platooning	Autonomous driving slice, infotainment slice	UAV-assisted 5G slicing for V2V applications	RAN slicing for autonomous tram
Number of involved BSs and clients	BS: 11; Client: 2000-4000	BS: 1; Client: 1	BS:1; Client: N.A.	BS: 1; Client: 9 + 9 UAVs	BS: 21; Client: 63
Performance metrics	Coverage ratio, connection ratio, occupied resources, handover ratio, blocking ratio	Connectivity, latency, packet delivery ratio	Throughput, packet reception ratio	Throughput, packet transfer delay, jitter, packet loss ratio	Packet delivery ratio, required bandwidth with guaranteed delay
Algorithm complexity	$O(n_c n_s)$	N.A.	N.A.	N.A.	N.A.
Mathematical analysis	N.A.	N.A.	SINR analysis	Fuzzy decision-making, optimization	Optimization
Implementation tool	SliceSim, Python 3.7	C, GeoNetworking, open-source MANO	System level simulations	SUMO, ns3	LTE simulator written in C++
Real-life or simulation	Real-life scenario plus simulations	Field trial	Simulation only	Simulation based on campus map	5G-air simulator

‡ Abbreviations used in this table: n_c – number of clients covered by a BS and n_s – number of slices offered by a BS; MANO – management and orchestration; SINR – signal-to-interference-plus-noise ratio; ns3 – network simulator v3.

Different from the aforementioned related work, our work proposes a framework to analyze possible deploying scenarios in a real-life area of interest by taking into consideration all the slices with a main focus on the V2X slice. We analyze performance parameters for every case and give insights on how network slicing should be used in areas where heterogeneous services are requested. In Table 1, we make a qualitative comparison of the proposed framework in this paper versus a few existing network slicing solutions dedicated to V2X applications. Furthermore, it is worth mentioning that usage control is an effective procedure for safety decidability and such techniques may apply to V2X applications [24][25][26].

3. Network slicing: architecture and slices

In this section, we first present a general architecture for network slicing and then introduce the network slices considered in this study.

3.1. Network slicing architecture

Enabled by the integration of SDN and NFV, network slicing is facilitated on a partially shared infrastructure. The common infrastructure contains two types of hardware, one being the dedicated hardware for radio access networks (RANs) and the other being the generic shared hardware for NFV infrastructure resources. The network functions that work on shared hardware are customized based on the requirements of each slice but they cannot be applied in cases where dedicated hardware is compulsory.

In terms of deployment scenarios, two types of network slicing are present in 5G networks:

- Slicing for QoS: Creation of slices when the QoS requirements are taken into consideration for specific applications of end users. This type of slicing is built based on sharing services with a higher priority for certain types of services, e.g., video streaming, gaming, medical emergency use cases.
- Slicing for infrastructure sharing: The ability to share the entire infrastructure of a slice, including at the RAN level. If an operator that owns a slice gives access and shares its infrastructure among its users, the slice could be easier to modify and adapt since its structure can be changed by tenants.

According to [27], the network slicing concept consists of three functional layers, as depicted in Fig. 3.1.

- Service Instance Layer: It represents services demanded by business companies or end users. The services can be provided by the operator or a third party and every service is represented by a service instance.
- Network Slice Instance Layer: It provides network characteristics required by the Service Instance Layer. In order to create a Network Slice Instance, the operator uses a Network Slice Blueprint, which is basically a list of physical and logical requirements, a complete description of the structure, configuration and the workflows of how to instantiate and control the Network Slice Instance.
- Resource Layer: It refers to the physical and virtual network functions used to implement a slice instance. At this layer, the partitioning of the resources is managed based on NFV orchestration and application resource configuration.

5G technologies aim to provide an end-to-end infrastructure that is able to deliver high QoS in an environment across a variety of use cases. Network slicing has a wide range of applicability in real-life scenarios and can enhance network performance as well as user experience. By dividing a set of applications, each slice can be responsible for a specific type of services with requirements that are different from other slices.

3.2. Network slices

The services requested by end users considered in this study contain four use cases, namely, eMBB, URLLC, mMTC, and V2X.

3.2.1. Enhanced mobile broadband

eMBB is characterized by broadband data access in specific locations such as crowded areas like stadiums, shopping centers, or uni-

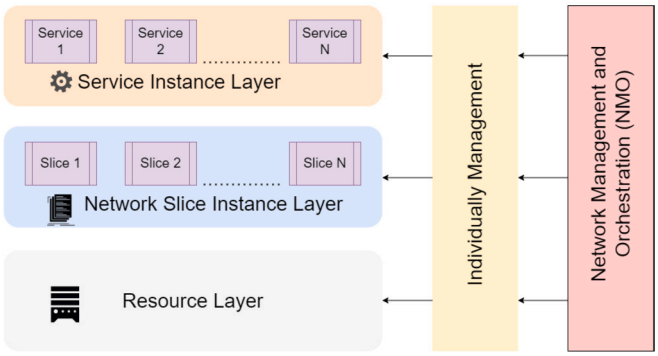


Fig. 3.1. Network slicing architecture.

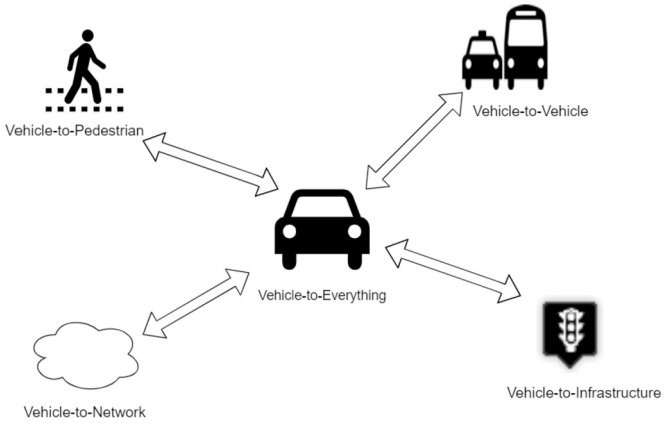


Fig. 3.2. Illustration of V2X communications.

versity campuses. It also assures connectivity over a large coverage or at high-speed for public transportation [28]. In brief, eMBB focuses on improved data rates (around 10 Gbps or higher).

3.2.2. *Ultra-reliable low latency communications*

This type of communications, as the name suggests, requires high speed with an ultra-high probability of successful transmissions. This use case is mainly targeted at industrial automation, drones and robots, augmented reality (AR), virtual reality (VR), and emergency-related scenarios.

3.2.3. *Massive machine-type communications*

Internet of things (IoT) devices are characterized of low cost, low traffic intensity, and long-lasting battery lifetime. mMTC needs to support a high density of devices in urban, extra urban, and rural areas, where a massive number of sensors, meters, cameras etc. may exist. These devices are deployed for various purposes, varying from managing lightning infrastructure in cities and countryside, for measuring noise, pollution, and temperature levels, etc. [29].

3.2.4. *Connected vehicles and V2X*

Connected vehicles represent a special use case in 5G networks since more and more cars are fully electrical, with a lot of IoT devices or sensors that require network connectivity. A combination of the requirements for eMBB and URLLC is a perfect deployment scenario for connected vehicles. This category of use cases supports advanced safety applications which mitigate road accidents, improve traffic efficiency, prevent traffic jams, and improve the access for emergency vehicles. These applications require a dedicated slice with features supporting low latency for warning signals, high data rates to share sensor data, as well as information sharing between vehicles and infrastructure.

As defined by the 3rd generation partnership project (3GPP), V2X consists of four types of communications, namely, V2V, V2I, vehicle-to-network (V2N), and vehicle-to-pedestrian (V2P) (shown in Fig. 3.2). V2X communications require 5G networks to accommodate the following capabilities [30]:

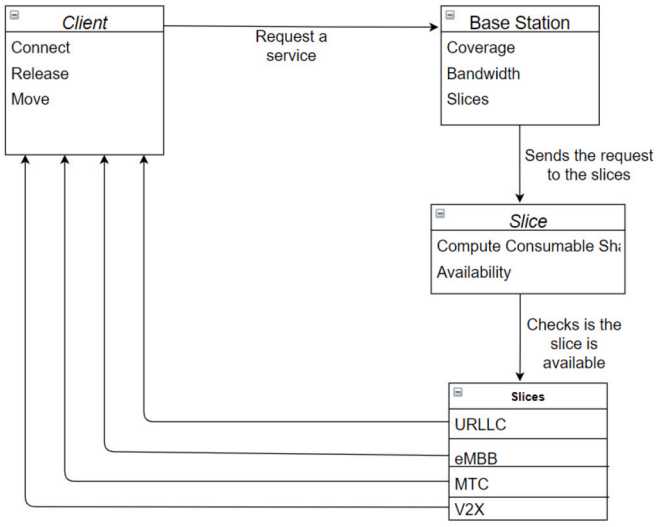


Fig. 4.1. Overview of the proposed network slicing framework.

- Speed – Peak data rate can hit 20/10 Gbps for downlink/uplink respectively per base station.
- Latency – 4 ms in typical conditions and 1 ms for use cases that demand the upmost speed.
- Capacity – 5G should be able to support one million connected devices per square kilometer.

4. Proposed network slicing framework

In this section, we first give an outline of the proposed network slicing framework and then present detailed algorithms for the three slicing instances defined in our framework.

4.1. *Framework overview*

In Fig. 4.1, we present a sketch of the proposed network slicing framework for V2X applications considering the co-existence of eMBB, mMTC, and URLLC traffic. Within this framework, we have defined there components or network slicing instances as outlined below. The three slice instances need to inter-operate and communicate with each other as shown in this figure.

- *Client:* A Client instance that needs to be installed in each vehicle.
- *Base Station:* The Base Station instance is responsible for receiving requests from clients and forwarding them to the Slice Management instance.
- *Slices* or *Slice Management:* The Slice instance checks the availability of radio resources and allocates accordingly resources to each slice.

When operating the framework, a Client instance initiates a service request towards the closest base station. At that moment, the Base Station, which has a specified coverage, bandwidth, and a set of allocated slice resources, sends a request to the Slice Management instance checking whether the requested slice has sufficient resources. The Slice Management instance computes the consumable share and checks the availability of the requested slice resources. If the slice can deliver the requested services to the UE, corresponding resources are allocated to the respective client. Then a slice which could be URLLC, eMBB, mMTC, or V2X will be in charge of making a connection to the client.

4.2. Algorithms for the three slice instances

In this subsection, we present in details the algorithms that need to be performed for each slice instance.

4.2.1. Client

The Client instance is the one that represents end users in a network (UEs, devices, vehicles, etc.). There are four steps that a client needs to make in each cycle of a connection session:

- Lock
- Stats
- Release
- Move

Locking represents the moment that a client connects to the closest base station and starts to consume or disconnects from it if is already connected. The pseudocode shown in Algorithm 1 describes the connect function in the locking step and counts the number of connected users to a base station. The handover counter increases when a handover is made and the blocking counter increases when a request is blocked (when no sufficient resource available in the slice).

After that, the client has to release the resources it occupied before it disconnects in order for the next client(s) to have available resources as shown in Algorithm 2.

Algorithm 1 Connect function

Input: Slice
Output: Connected clients in coverage

if *Client* is connected **then**
  Return;
end if
Connect attempt ++;
if *Slice* is available **then**
  Connected users ++;
  Connected ← *True*;
  return True;
else
  Assign closest basestation;
  if *Basestation* exists *and slice* is *available* **then**
    Handover count ++;
    if Base station exists and slice is not available **then**
      Block count ++;
    else
      print 'Client is not in coverage'
    end if
  end if
end if

Algorithm 2 Release consume function

Input: Client
Output: Release_consume()

if *Client* is *connected* and *last usage* > 0 **then**
  Release _ consume();
  if *remaining usage* <= 0 **then**
    disconnect;
  end if
end if

After resources are released, the client can be disconnected from the base station. The algorithm for this step is shown in Algorithm 3.

The last step of a connection cycle is the movement of the client and this procedure is presented in Algorithm 4. After establishing a connection with a base station and consuming the resources allocated to each requested slice, the clients will be disconnected and they may change their positions in space to make a new connection.

To identify the placement of clients and base stations in a region of interest, coordinates are needed. While base stations deployed on fixed locations, and the position of each client is random and may change due to mobility.

Algorithm 3 Disconnect function

Input: Client is connected
Output: Disconnected client

if *connected* == *False* **then**
  print 'Client is already disconnected from this slice';
else
  get slice;
  connected users ← *connected users* − 1;
  Connected ← *False*;
  print 'Client connected to this slice';
  return *not connected*;
end if

Algorithm 4 Movement of the client

Input: Initial position of the client (x,y)
Output: Final position of the client

x, y take the mobility pattern and generate movement;
x ← *x* + *x*;
y ← *y* + *y*;
if *Base station exists* **then**
  if *client is not in coverage* **then**
    disconnect;
    assign closest base station;
  end if
else
  assign closest base station;
end if

Mobility pattern represents the random distribution of client locations. Various mobility models and UE distributions may apply in real-life networks. However, to study mobility model or client position distribution is beyond the scope of this paper. Once the coordinates (x,y) of the current position of a client are known, we are able to check whether it is within the coverage of a base station. If yes, the closest base station to that client is assigned and the checking for the requested slice could begin.

4.2.2. Base station

There are multiple base stations in a network. Each base station only stores the data related to base stations from a configuration file and sends it to other instances. The most important specifications of a base station in this kind of algorithm are the index of the base station, its coverage (to know if a client can be assigned to it or not), the capacity of the base station in terms of radio resources and the slices that are contained at each base station. Different BSs may have different distribution of radio resources for every slice.

In our framework, the base station script has a very simple function because it does not perform any kind of operation by itself. A BS is a class described by its position, coverage, capacity and what slices it contains. We assume that it is in the broadcast mode all the time, with clients connected and disconnected from it.

4.2.3. Slice management

The slice management instance is responsible for the verification of the availability of the requested resource blocks. It contains information regarding to all the slices at a BS, such as the slice identities, the resource blocks occupied, connected UEs etc. Some slices, such URLLC where ultra-low latency or reliability receives privileged priority, may have a set of guaranteed resource blocks that is delivered regardless of the traffic load status. On the other hand, some mMTC applications may not have guaranteed resources if they are delay tolerant. Another type of information, aside the functions that will be discussed and where we have contributed, is the QoS and the delay tolerance of a slice.

Slices are contained at a BS, and each slice receives a portion of resources indicating how much of the total capacity the BS has allocated to it. Some slices, like eMBB, which require high data rates need a higher number of resource blocks. Other slices, e.g., mMTC applications

Algorithm 5 Slice management class

Input: name, ratio, connected users, client weight, delay, QoS class, guaranteed bandwidth, maximum bandwidth, initial capacity;
Output: Selected slice

Define the used parameters: name, ratio, connected users, client weight, delay, QoS class, guaranteed bandwidth, maximum bandwidth, initial capacity;
def function get_consumable_share:
if connected users == 0 **then**
    return min(initial capacity, maximum bandwidth;
else
    return min(initial capacity / connected users, maximum bandwidth);
end if
def function is_available:
$real\ capacity \leftarrow min(initial\ capacity,\ maximum\ bandwidth)$;

$next\ bandwidth\ for\ use \leftarrow real\ capacity/(connected\ users + 1)$;

$actual\ QoS\ of\ slice = floor(delay * client\ weight)$;
if *next bandwidth for use < guaranteed bandwidth* ***and*** *actual QoS of slice < QoS class* **then**
    return $True$;
else
    return $False$;
end if

that deliver services for IoT devices, need a smaller amount of resource blocks. For slice management, we define a ratio of resource blocks in the configuration file indicating the amount of resource blocks allocated to a slice. The higher the ratio, the larger the amount of resources to a slice, and vice versa.

Algorithm 5 outlines the pseudocode for slice management including determining the QoS class and the corresponding functions. The function that computes the consumable share of each slice is *get_consumable_share* and it obtains the minimum value between the initial capacity of the slice (which is calculated according to the ratio for each slice) and the maximum amount of resources allocated to each slice. This step is done when no client is connected. When clients start to connect, the initial capacity decreases accordingly, since the total capacity is divided by the number of connected clients. The next function is to check whether the requested slice is available and this step is doing with respect to cell capacity and QoS requirements.

Furthermore, the slicing management class has a parameter referred to as *client weight*. This is another important function when determining the QoS of a connection. While a higher weight represents enhanced services for instance for a phone conversation, a lower weight represents other normal or lower priority types of services with lower QoS requirements. One idea introduced in this work is to calculate the actual QoS level of every slice depending on delay tolerance as well as client weight and compare it with a predefined threshold. More specifically, the actual QoS of a slice is the product of delay tolerance and client weight, and it is compared with the QoS class, which is a parameter predefined in the configuration file. If the actual QoS of the slice is lower than or equal to the QoS class, the respective slice has a higher priority. Aside from this check, the algorithm verifies whether the residual capacity after existing clients is connected is enough to sustain more UEs. If these two conditions are fulfilled, then the slice is available and can be delivered to an incoming client that requests services.

4.3. Overall flowchart

In addition to these three essential instances (client, base station, and slice management) presented above, a few other functions are also needed in order to operate the framework. Such additional functions include to distribute the clients in space, compute the shortest distance, and calculate the final results. Furthermore, all the resources are maintained in a container so that they can easily be allocated according to the requested resources or released back when the occupied resource blocks are no longer required.

In Fig. 4.2, we illustrate the flowchart of the framework including all the functions that have been explained. The procedure starts from

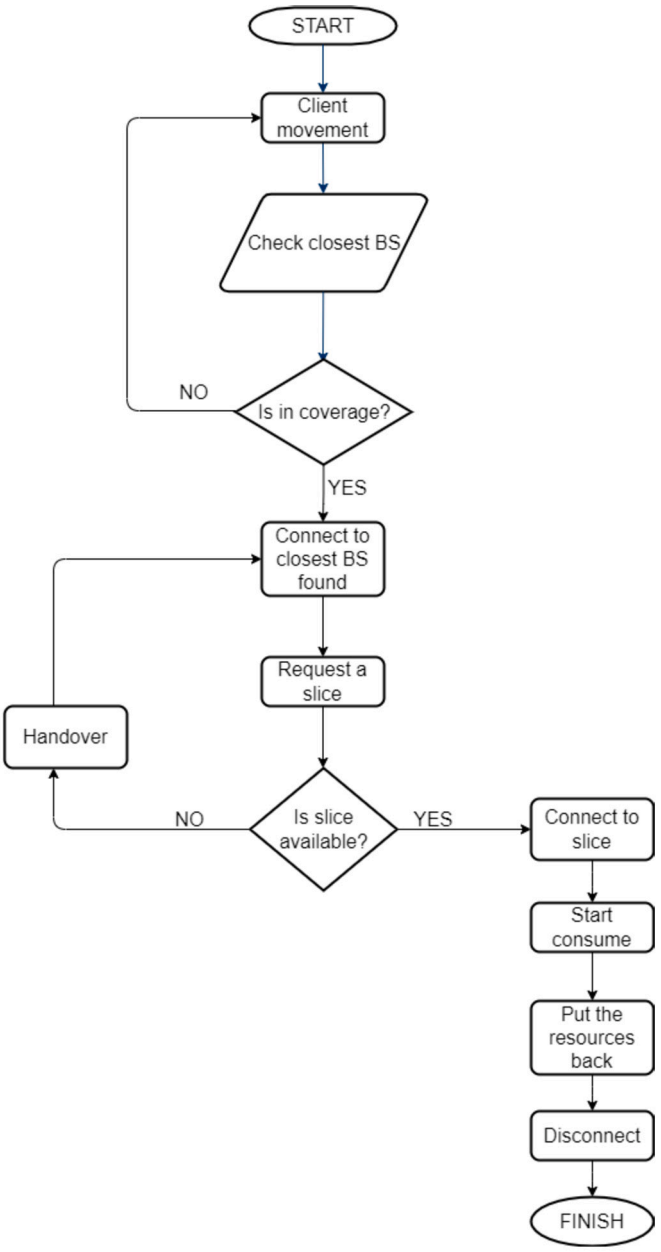


Fig. 4.2. Program flowchart.

the movement of a client and it checks whether the client is covered by a BS. If yes, the client is assigned to the closest BS. If the UE is not in the BS coverage from the start or if it is outside the coverage before movement, a handover is needed. Once the client is within the coverage, it can connect to the BS and request a slice. Here the slice management function comes into play and it checks the availability of the requested resources. If no resource is available with the slice, a handover is made. In this case, the type of handover is of inter-cell type, because it connects the client to another BS.

However, if the corresponding slice is available only in terms of capacity and not in terms of QoS, the new request will be blocked. Once all the conditions are fulfilled and the slice is available, the client will be successfully connected to that slice and it starts to consume resources. After the consumption is made and the client no longer needs resources from that slice, the occupied resources must be released, available for consumption by other clients. As the last step, the client can be disconnected from the BS and it eventually begins a new connection to another BS.

Fig. 4.3. Real-life positions of 11 base stations in an area of interest in the city of Bucharest.

Table 2
Slice distributions across four slices at 11 base stations

BS ID	BS capacity	eMBB [%]	URLLC [%]	mMTC [%]	V2X [%]
922501	25 Gbps	48	30	20	2
910661	30 Gbps	0.3	0.3	0.3	0.1
945911	30 Gbps	20	30	10	40
927241	25 Gbps	45	2	15	38
933301	20 Gbps	45	50	3	2
938721	20 Gbps	14	5	45	36
937121	40 Gbps	28	50	20	2
934101	30 Gbps	47.9	32	2	0.1
933791	30 Gbps	40	30	25	5
934701	20 Gbps	20	55	10	15
927941	35 Gbps	50	5	25	20

Table 3
Flow specifications w.r.t. QoS and data rate requirements

Slice	Delay tolerance	QoS	Guaranteed capacity	Max. capacity
eMBB	2	5	100 Mbps	1 Gbps
mMTC	10	2	1 Mbps	10 Mbps
URLLC	1	2	5 Mbps	50 Mbps
V2X	1	1	5 Mbps	100 Mbps

5. Implementation and network configurations

In this section, we first give an outline on the implementation of the algorithms presented in Section 4 and then present in details the network configurations of our simulation based performance assessment. As a case study, our network scenarios are built based on real-life base station deployment by a network operator, Vodafone Romania, in the city of Bucharest, in an area nearby the campus of the University Politehnica of Bucharest (UPB).

5.1. Implementation outline

The pseudocodes for all the algorithms presented above have been implemented based on a popular simulation platform for 5G networks [31]. The functions for clients, BSs, and slice management are written in Python 3.7. using different libraries like Simpy, KDTree, and Matplotlib. To assess the performance of the proposed framework, the client instance has to be implemented in all clients/UEs and the BS and slice management instances have to be implemented in all base stations.

As the core of the whole framework, we have created a slicing function that checks whether a service is available or not and distributes the slice resources across the whole network. We have also implemented all the new slices with new specifications that fit for this type of network.

Furthermore, the QoS parameters needed in Algorithm 5 are affected by the delay of every service that slices are delivering and are not depending only on the allocated or available capacity for every slice. When delay becomes significant, the achieved QoS will be poor, and vice versa. The QoS levels defined in this algorithm have 5 values, with 5 being the poorest service and 1 representing the best one. For delay tolerance, we consider 10 levels where 10 means most delay tolerant and 1 stands for most delay sensitive.

5.2. Scenarios and network configurations

For performance assessment, we have identified a few use case scenarios based on the real-life positions of 11 base stations deployed close to the UPB campus as presented in Fig. 4.3. This region has been selected as an area of interest because there is a big flow of clients and vehicles and it is also a big commercial zone in the city. In this figure, the gNBs are represented by the green circles and for each one of them the position is represented according to the real placement and its coverage is approximated with a circle for the sake of simplicity. The gNBs have the coordinates in the center of the circle (marked by the green circles) with different coverage ranges (as shown later in Figs. 5.1 and 5.2) respectively.

The cell capacity at each cell (in Gbps) is partitioned into four slices, each dedicated to one type of applications, i.e., eMBB, URLLC, mMTC, or V2X. The slice resource allocations for the four slices across 11 gNBs are illustrated in Table 2. This slice distribution is represented by a pre-configured ratio in the configuration file and it dictates how much percent of the total capacity of each cell is allocated to every slice. Note that this ratio is re-configurable.

The values for the distribution of these slices at various gNBs are configured by taking into consideration the positions of the gNBs in the area of interest as well as their coverage. For example, for a base station that is closer to a road, the V2X slice ratio increases and for another one that is closer to an office building or shopping center, the eMBB ratio is higher.

Even if each slice has an initial capacity that depends on the ratio configured by the base station, a guaranteed capacity and a limit maximum capacity have been introduced for every slice. With guaranteed capacity, an existing traffic flow belonging to a low priority class will sacrifice its data rate when more high priority traffic flows are connected but *it will not get starved*. With maximum capacity, a high priority traffic flow cannot increase its data rate (which leads to better QoS) to higher than this value so that there are still resources available for other flows. The specifications for the slices are the presented in Table 3. The higher the level (with a smaller value for delay tolerance and QoS), the more important the service. For example, with the level being 1 for both delay and QoS, the V2X slice will receive the highest priority.

To investigate the performance of the proposed framework under heterogeneous traffic classes, we introduce a configurable parameter referred to as *client weight*, denoted by α, which dictates the distribution of clients across four categories of services. Note that the sum of all the weights must be 1. As an example, the weights for one of our simulations are illustrated in Table 4 showing that among all clients 5% of them belong to V2X users, etc. For other simulations, these weights are reconfigured to different values in order to explore the impact of the V2X slice on network performance.

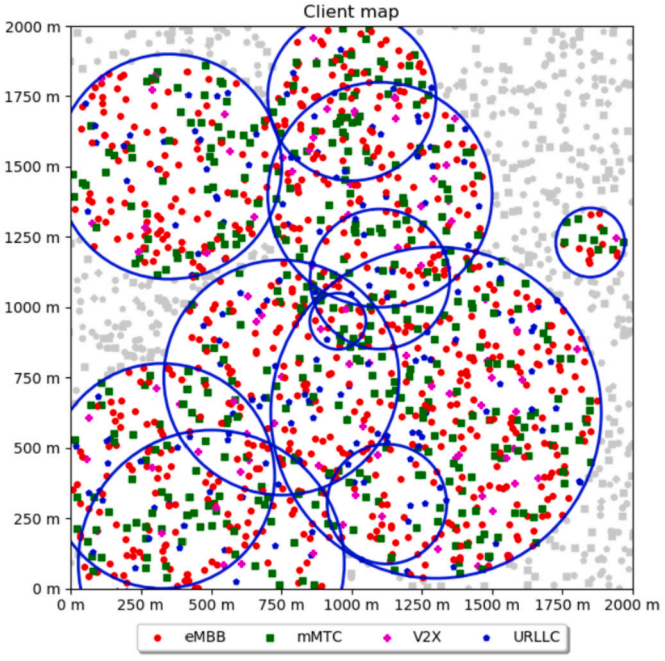


Fig. 5.1. Client distribution map for 2000 clients.

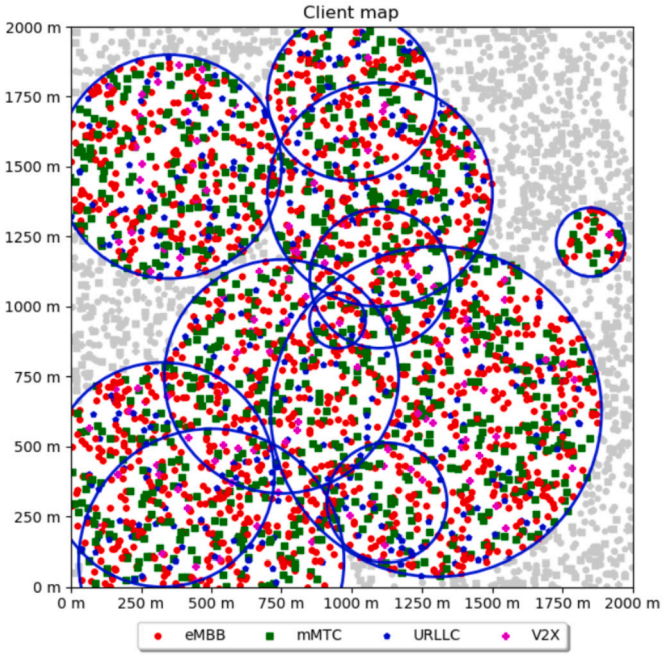


Fig. 5.2. Client distribution map for 4000 clients.

Table 4
Client weights for each slice

Slice	Client weight
V2X	$\alpha = 0.05$
URLLC	$\alpha = 0.13$
mMTC	$\alpha = 0.32$
eMBB	$\alpha = 0.50$

Table 5
Connectivity behavior: 2000 versus 4000 clients

Number of clients	2000	4000
Average coverage ratio	0.54	0.50
Average connection ratio	0.42	0.36
Average occupied resource (Gbps)	22.578	23.931
Average blocking ratio	0.1980	0.2201
Average handover ratio	0.0125	0.0077

5.3. Client distribution map and density

Considering that the region of interest has a total area size of 4 square kilometers, we configure the total number of clients in our network as 2000 and 4000 clients respectively. The clients are randomly distributed following the uniform distribution across the whole region of interest, as shown in Figs. 5.1 and 5.2 respectively. In the next section, we present our simulation results based on the network configurations and client distributions explained in this section.

6. Simulations and numerical results

In this section, we present the obtained simulation results based on the network configurations explained in Section 5. For each simulation run, we emulate service requests happened within a duration of one hour and the reported results in this section are the average values obtained from multiple simulation runs and are averaged over the values obtained across all the 11 BSs.

In this study, the following five metrics are defined for network performance evaluation. As the numerical results reported hereafter are obtained based on the trace files created through our simulations, we do not introduce any mathematical symbols or expressions for these metrics.

- Average coverage ratio: Defined as the area that is covered by the base stations with respect to the entire area of the interest.
- Average connection ratio: Defined as the number of clients that are successfully connected divided by the total number of clients in the network.
- Average occupied resources: Defined as the total amount of network capacity (in Gbps) used throughout the whole duration.
- Average handover ratio: Defined as the percentage of ongoing connections that have to perform handover.
- Average blocking ratio: Defined as the percentage of connection requests that have been blocked due to lack of resources.

In what follows, we first configure the client weight as $\alpha = 0.05$ and investigate in Subsection 6.1 the network performance in terms of all five metrics. Then in Subsections 6.2 and 6.2, we explore the performance with respect to the last three metrics for two client populations (with 2000 and 4000 clients respectively) by varying α when the percentage of clients that require V2X services increases.

6.1. Overall performance: 2000 versus 4000 clients

When there are 2000 or 4000 clients distributed in the area of interest, the network performance in terms of the five defined metrics is illustrated in Table 5. Note that the size of the area of interest and the total network capacity are the same even though there are different number of clients distributed in this area.

As expected, more resources are occupied when the number of clients is larger. While 36% of the clients are connected in the 4000-client case, the successful connection ratio for the 2000-client case is only 42%. These values seem to be a low in both cases, but the reason behind this behavior is that a large number of clients being located outside the coverage of any base station (the average coverage ratio being

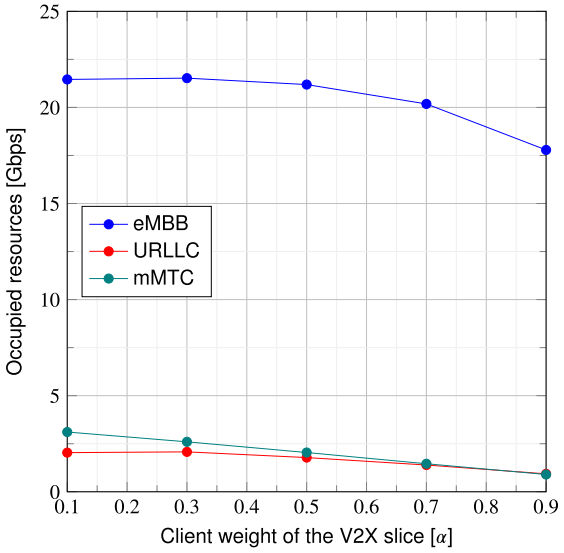


Fig. 6.1. Occupied resources for 2000 clients.

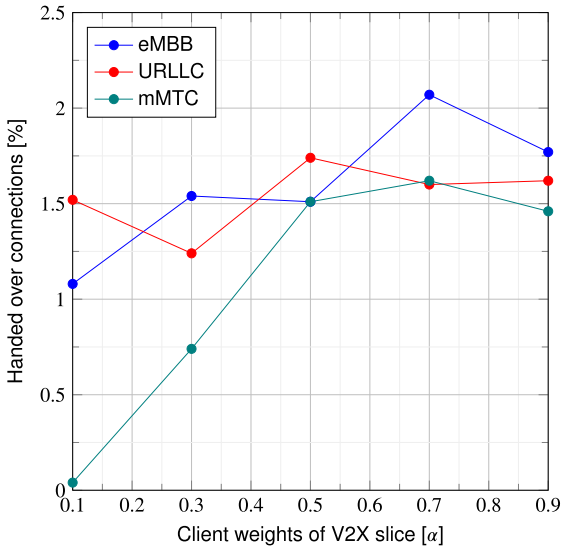


Fig. 6.2. Handover ratio for 2000 clients.

54% for the 2000-client case and 50% for the 4000-client case). In both cases, there are not sufficient radio resources to connect more clients.

6.2. Network performance: 2000 clients with a varying number of V2X users

To test the impact of the V2X slice on network performance, we reconfigure the client weight for the V2X slice to various values as $\alpha = [0.1; 0.3; 0.5; 0.7; 0.9]$ respectively. Since the sum of the weights for all slices has to be 1, the other slices (URLLC/eMBB/mMTC) will have a weight of $1 - \alpha$. All the results shown in this and the next subsection are displayed according to the client weight of the V2X slice with respect to the total weight.

6.2.1. Occupied resources

The curves illustrated in Fig. 6.1 represent the consumed resources obtained as an output of our simulations. In all figures presented in this subsection, the *Y* axis represents the occupied resources in Gbps and the *X* axis indicates the client weight of the V2X slice.

As can be observed in the figure, the eMBB slice consumes the largest portion of the total radio resources but it its occupancy decreases from ~21 Gbps downwards to ~17.5 Gbps as more V2X users arrive. This is because the eMBB slice requires requires large capacity in comparison with the other slices. The URLLC slice occupies the smallest chunk of resources since it represents services that do not require much radio resource but with stringent low latency and high reliability requirements. Even though the resource consumption of the eMBB/URLL/mMTC slices decreases when the weight for V2X increases, the V2X slice does not have strong impact on the consumed resources. This is because the required capacity for V2X services is generally low in comparison with the other categories of services.

6.2.2. Handover ratio

Handover represents a process in which an ongoing transmission (in this case mobile data) is transferred from one cell to another in order to maintain connectivity and to keep the link up until the connection is terminated. Fig. 6.2 illustrates how the handover ratio of the other three slices is affected by the V2X slice as the V2X service has to assure high reliability and low latency and no V2X connection shall be interrupted for the safety of the users.

As the client weight for the V2X slice increases, the handover ratio tends to increase as well. This is because the V2X services are characterized by the fact that clients change their positions in space in a fast manner, leading to more data transfer from one BS to another. The lowest handover ratio is observed in the mMTC slice when the client weight for V2X reaches the smallest as 0.1. The reason is that, in this case, most of the clients are connected to the mMTC slice, which is represented by stationary devices. Higher handover ratios are observed in the other cases, where eMBB and URLLC slices help the V2X services to use their high specifications in terms of capacity, latency and reliability. Furthermore, it can be observed that the handover curves descend for the eMBB and mMTC slices when α goes up from 0.8 to 0.9. This is because as the number of V2X clients increases, the handover decreases since these are the slices that do not require high reliability as URLLC and V2X slices. Also, the blocking of the new connections tends to increase for α from 0.8 to 0.9 for the same reason, as it can be seen in Fig. 6.3.

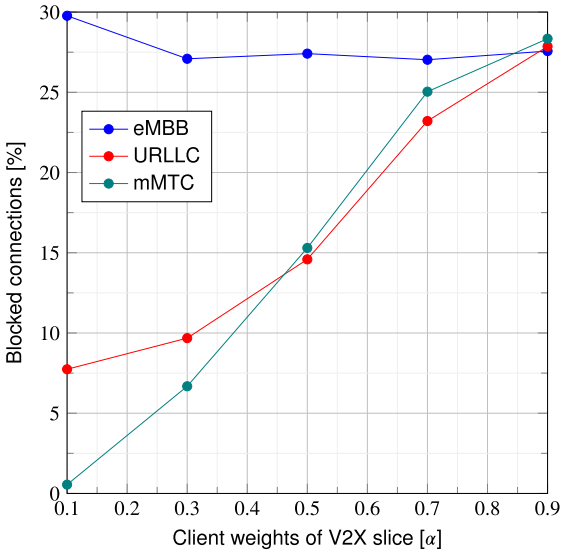


Fig. 6.3. Blocking ratio for 2000 clients.

6.2.3. Blocking ratio

Blocking happens to a new service request by a client if no sufficient network resource is available. Depending on the QoS requirements and resource availability, different criteria may apply for decision making. The blocking procedure helps to ensure QoS provisioning as already

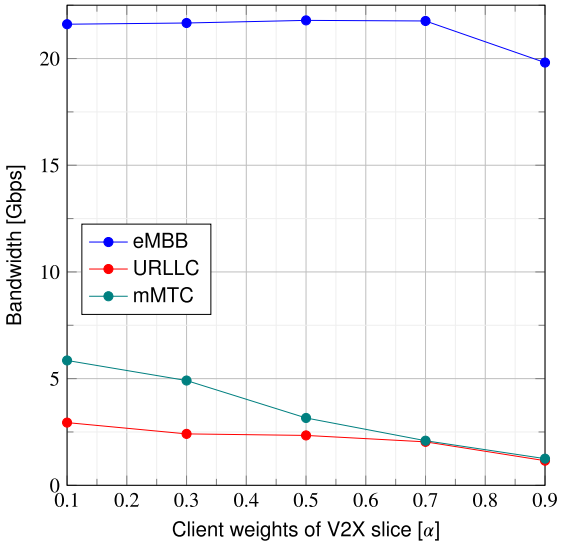


Fig. 6.4. Occupied resources for 4000 clients.

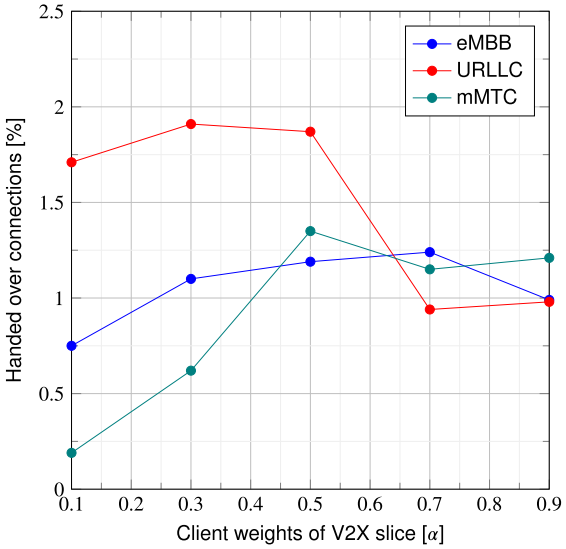


Fig. 6.5. Handover ratio for 4000 clients.

established connections will be maintained regardless of the admission or blocking decision of the new request.

As depicted in Fig. 6.3, the highest percentage of blocking is manifested by the eMBB slice from the beginning of the process. When the client weight for the V2X slice is the smallest ($\alpha = 0.1$) and the one for eMBB has the value of 0.9 as only two slices are configured in each set of our simulations.

The eMBB slice pushes the network to a high blocking ratio due to its high requirements in terms of high data rates and high reliability, leading to insufficient resources at gNBs to maintain the connections up. As the weight for the V2X slice increases, the blocking ratios for all the other service categories tend to stabilize around the same value of ∼27.5%. This result provides another insight on how a high priority slice is protected for QoS ensurance. As the V2X slice is deemed to have the highest priority (as shown in Table 3), blocking of new requests from other service categories is imposed in order to keep the already established connections secure with a very low rate of dropping.

6.3. Network performance: 4000 clients with a varying number of V2X users

To further investigate the behavior of the network, the case where 4000 clients are present in the area of interest is studied in this subsection. The same network configurations as used in Subsection 6.2 are applied and the performance is compared with the previous case with 2000 clients.

6.3.1. Occupied resources

The consumed radio resources are computed in the same manner as presented above and the results are illustrated in Fig. 6.4. Intuitively, the difference between the results obtained from the 2000-client and 4000-client cases should be larger, compared with the resource occupancy status observed in Fig. 6.1 and Fig. 6.4. This is because the density configured in both cases is already quite high, as approximately 180 and 360 clients per BS. When a larger number of clients need to be supported by each BS, the network becomes saturated. So deploying more clients will not help to increase total resource occupancy.

As shown in Table 3, the slice that is most resource demanding is the eMBB slice as high data rates are expected for eMBB applications. When the number of V2X is not very high (with a slice weight $\alpha = 0.7$ or less), the eMBB applications will consume most resources. The reason is that V2X does not need high data rates. However, when $\alpha > 0.7$ for V2X, the occupied resources for eMBB start to descend. This is because V2X applications have higher priority as explained earlier.

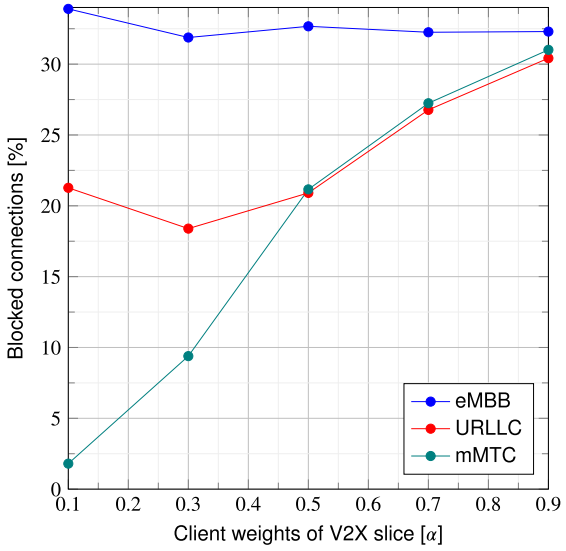


Fig. 6.6. Blocking ratio for 4000 clients.

6.3.2. Handover and blocking ratios

The obtained results for handover and blocking in the 4000-client case are presented in Fig. 6.5 and Fig. 6.6 respectively. Compared with the other two metrics (occupied resources and blocking ratio), the handover ratio in the 4000-client case has a lower value than the 2000-client case. This is because more new service requests are blocked when there are more clients in the network. When more UEs are present in an area, the network will be more likely congested. This is the reason why the blocking ratio is higher and the handover ratio is lower. With more users, a smaller percentage of the connections will be handed over to other base stations and a higher percentage of requests will be is blocked, due to the occupancy of network resources.

When the client weight of the V2X slice exceeds $\alpha = 0.5$, the handover ratios for URLLC and mMTC services start to decrease whereas their blocking ratios exhibit an opposite tendency. This is because the network intends to maintain ongoing connections since dropping already established connections leads to more serious QoS degradation than blocking new requests. Furthermore, it is worthy mentioning that for delay tolerant services blocking does not mean complete rejection of a client. Instead, the service request will stay in a buffer until a nearby base station has resources to deliver the requested service.

7. Further discussions

In this section, we further discuss a few aspects that are related to the framework presented above, from algorithm complexity, dynamic slice distribution (client weights), to potential real-life operation.

7.1. Algorithm complexity

To enable the framework, the proposed algorithms need to be installed at BSs and clients respectively. At a BS, the service requests and slice allocation are processed consecutively for each client based on the number of network slices. Therefore, the algorithm time complexity is in the order of $O(n_c n_s)$ where n_c and n_s stand for the number of clients covered by a BS and the number of slices offered by a BS respectively. At the client end, each client needs merely to maintain and update the four steps explained in Subsec. 4.2.1, i.e., Lock, Stats, Release, and Move. Therefore, the complexity of the algorithm to be executed at each client is exceedingly lightweight.

7.2. Dynamic slice distribution

The α values that are shown in Table 2 above are meant for this case study and they are manually configured in our simulations. *In real-life, these values should be dynamically adjusted based on BS capacity and measured instantaneous traffic load conditions.* Particularly, how to adaptively adjust client weight, i.e., α for each base station in the area of interest, in a real-time fashion is a challenging task, as traffic intensity varies from time to time. This task becomes even more challenging when slice distribution needs to be coordinated among multiple base stations as α varies among different base stations. However, to further study how to dynamically adjust slice distribution, for instance at a minute or second level, is beyond the scope of this paper.

7.3. Real-life implementation

The framework proposed in this paper fits better urban scenarios than rural ones as more types of services may prevail in urban scenarios. This study focuses on a real-life base station deployment scenario and the numerical results are obtained through simulations based on this scenario. To potentially implement our algorithms in real-life and operate them in a real-time manner, it requires substantial efforts from a mobile operator, such as network software upgrade and instantaneous traffic measurements. Nevertheless, we believe that the work presented in this study serves as a valuable reference towards real-life implementation of network slicing for V2X applications in the near future.

8. Conclusions and future work

In this paper, we have studied the effect of network slicing for V2X services on network performance by jointing considering the X2V slice with three other slices (eMBB, URLLC, and mMTC) in the same network consisting of multiple base stations. We proposed a framework for network slicing which consists of a set of algorithms needed for both clients and base stations. Based on a deployed real-life network topology, we investigate the performance of network slicing through extensive simulations, considering various network configurations and client distributions under heterogeneous service categories.

It has been demonstrated that, as the number of clients in the area of interest varies, network performance with respect to consumed resources, handover, and blocking fluctuates. To provide ideal performance under various network deployment and traffic conditions, resource allocations among slices must be balanced. As our future work, we will design network slicing schemes that take time-varying traffic and service conditions into account for dynamic resource allocation. More real-life scenarios will be identified at other areas of interest and more measurements will also be performed. Furthermore, machine learning based algorithms for slicing resource allocation will be another direction for our future work.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Frank Y. Li reports financial support was provided by EEA Grants. Ciprian Zamfirescua reports was provided by EEA Grants.

Data availability

The data that has been used is confidential.

Acknowledgement

The research leading to these results has received funding from the EEA NO Grants 2014-2021, under project contract no. 42/2021, RO-NO-2019-0499 - “A Massive MIMO Enabled IoT Platform with Networking Slicing for Beyond 5G IoV/V2X and Maritime Services (SOLID-B5G)”.

References

[1] C. Sexton, N. Marchetti, L.A. DaSilva, Customization and tradeoffs in 5G RAN slicing, IEEE Commun. Mag. 57 (4) (2019) 116–122, https://ieeexplore.ieee.org/document/8703477.

[2] S. Wijethilaka, M. Liyanage, Survey on network slicing for Internet of things realization in 5G networks, IEEE Commun. Surv. Tutor. 23 (2) (2021) 957–994, https://ieeexplore.ieee.org/document/9382385.

[3] A.A. Barakabitze, A. Ahmad, R. Mijumbi, A. Hines, 5G network slicing using SDN and NFV: a survey of taxonomy, architectures and future challenges, Comput. Netw. 167 (2020) 106984, https://www.sciencedirect.com/science/article/pii/S1389128619304773.

[4] M.H. Castañeda Garcia, A. Molina-Galan, M. Boban, J. Gozalvez, B. Coll-Perales, T. Şahin, A. Kousaridas, A tutorial on 5G NR V2X communications, IEEE Commun. Surv. Tutor. 23 (3) (2021) 1972–2026, https://ieeexplore.ieee.org/document/9345798.

[5] A. Alalewi, I. Dayoub, S. Cherkaoui, On 5G-V2X use cases and enabling technologies: a comprehensive survey, IEEE Access 9 (2021) 107710–107737, https://ieeexplore.ieee.org/document/9497103.

[6] S. Gyawali, S. Xu, Y. Qian, R.Q. Hu, Challenges and solutions for cellular based V2X communications, IEEE Commun. Surv. Tutor. 23 (1) (2021) 222–255, https://ieeexplore.ieee.org/document/9217500.

[7] C. Campolo, A. Molinaro, V. Sciancalepore, 5G network slicing for V2X communications: Technologies and enablers, Chap. 13, in: Radio access network slicing and virtualization for 5G vertical industries, Wiley, 2021, pp. 239–257.

[8] E. Borcoci, A.-M. Drăgulinescu, F.Y. Li, M.-C. Vochin, K. Kjellstadli, An overview of 5G slicing operational business models for Internet of vehicles, maritime IoT applications and connectivity solutions, IEEE Access 9 (2021) 156624–156646, https://ieeexplore.ieee.org/document/9615227.

[9] L. Zhang, A. Farhang, G. Feng, O. Onireti (Eds.), Radio access network slicing and virtualization for 5G vertical industries, Wiley-IEEE Press, 2021, https://www.wiley.com/en-ie/Radio+Access+Network+Slicing+and+Virtualization+for+5G+Vertical+Industries-p-9781119652380.

[10] I. Afolabi, T. Taleb, P.A. Frangoudis, M. Bagaa, A. Ksentini, Network slicing-based customization of 5G mobile services, IEEE Netw. 33 (5) (2019) 134–141, https://ieeexplore.ieee.org/document/8863735.

[11] T. Taleb, I. Afolabi, M. Bagaa, Orchestrating 5G network slices to support industrial Internet and to shape next-generation smart factories, IEEE Netw. 33 (4) (2019) 146–154, https://ieeexplore.ieee.org/document/8612449.

[12] I. Afolabi, J. Prados-Garzon, M. Bagaa, T. Taleb, P. Ameigeiras, Dynamic resource provisioning of a scalable E2E network slicing orchestration system, IEEE Trans. Mob. Comput. 19 (11) (2020) 2594–2608, https://ieeexplore.ieee.org/document/8772218.

[13] Y. Ren, T. Phung-Duc, J.-C. Chen, F.Y. Li, Enabling dynamic autoscaling for NFV in a non-standalone virtual EPC: design and analysis, IEEE Trans. Veh. Technol. 72 (6) (2023) 7743–7756, https://ieeexplore.ieee.org/document/10018535.

[14] S. Agarwal, F. Malandrino, C.F. Chiasserini, S. De, VNF placement and resource allocation for the support of vertical services in 5G networks, IEEE/ACM Trans. Netw. 27 (1) (2019) 433–446, https://ieeexplore.ieee.org/document/8611305.

[15] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, V.C.M. Leung, Network slicing based 5G and future mobile networks: mobility, resource management and challenges, IEEE Commun. Mag. 55 (8) (2017) 138–145, https://doi.org/10.1109/MCOM.2017.1600940.

[16] A. Lekidis, F. Bouali, C-V2X network slicing framework for 5G-enabled vehicle platooning applications, in: Proc. IEEE VTC-Spring, 2021, pp. 1–7, https://ieeexplore.ieee.org/document/9448769.
[17] W. Duan, J. Gu, M. Wen, G. Zhang, Y. Ji, S. Mumtaz, Emerging technologies for 5G-IoV networks: applications, trends and opportunities, IEEE Netw. 34 (5) (2020) 283–289, https://ieeexplore.ieee.org/document/9136587.
[18] H. Khan, P. Luoto, M. Bennis, M. Latva-aho, On the application of network slicing for 5G-V2X, in: Proc. European Wireless, 2018, pp. 1–6, https://www.vde-verlag.de/proceedings-en/564560007.html.
[19] S.A. Hakeem, A.A. Hady, H. Kim, 5G-V2X: standardization, architecture, use cases, network-slicing, and edge-computing, Wirel. Netw. 26 (2020) 6015–6041, https://link.springer.com/article/10.1007/s11276-020-02419-8.
[20] B. Ravi, M. Kumar, Y.-C. Hu, S. Hassan, B. Kumar, Stochastic modeling and performance analysis in balancing load and traffic for vehicular ad hoc networks: a review, Int. J. Netw. Manag. (2023) 1–22, https://doi.org/10.1002/nem.2224.
[21] S.K. Arora, G. Kumar, M. Hedabou, E.M. Amhoud, C. Iwendi, Blockchain-inspired lightweight trust-based system in vehicular networks, Int. J. Netw. Manag. (2023) 1–16, https://doi.org/10.1002/nem.2226.
[22] E. Skondras, E.T. Michailidis, A. Michalas, D.J. Vergados, N.I. Miridakis, D.D. Vergados, A network slicing framework for UAV-aided vehicular networks, Drones 5 (3) (2021) 70, https://www.mdpi.com/2504-446X/5/3/70.
[23] D. Tamang, S. Martiradonna, A. Abrardo, G. Mandó, G. Roncella, G. Boggia, Architecting 5G RAN slicing for location aware vehicle to infrastructure communications: the autonomous tram use case, Comput. Netw. 200 (2021) 108501, https://www.sciencedirect.com/science/article/pii/S1389128621004400?via%3Dihub.
[24] P.V. Rajkumar, R. Sandhu, Safety decidability for pre-authorization usage control with identifier attribute domains, IEEE Trans. Dependable Secure Comput. 17 (3) (2020) 465–478, https://ieeexplore.ieee.org/document/8362972.
[25] P.V. Rajkumar, R. Sandhu, Safety decidability for pre-authorization usage control with finite attribute domains, IEEE Trans. Dependable Secure Comput. 13 (5) (2016) 582–590, https://ieeexplore.ieee.org/document/7097658.
[26] P.V. Rajkumar, R. Sandhu, Security enhanced administrative role based access control models, in: Proc. ACM SIGSAC Conf. Comput. Commun. Security, 2016, pp. 1802–1804, https://dl.acm.org/doi/10.1145/2976749.2989068.
[27] P. Hedman (Ed.), Description of Network Slicing Concept, NGMN 5G Deliverable, 2016, https://ngmn.org/wp-content/uploads/160113_NGMN_Network_Slicing_v1_0.pdf.
[28] 5G Americas, Understanding mmWave spectrum for 5G networks, 5G Americas White Paper, 2020, https://www.5gamericas.org/wp-content/uploads/2020/12/InDesign-Understanding-mmWave-for-5G-Networks.pdf.
[29] 3G4G Ltd, 5G Spider Diagrams, https://www.slideshare.net/3G4GLtd/misc-5g-spider-diagrams, 2019.
[30] Thales Group, eSIMs to 5G: how connected cars stay connected, https://www.thalesgroup.com/en/markets/digital-identity-and-security/iot/inspired/connected-cars/esim-to-5G, 2020.
[31] M.E. Güre, SliceSim: a simulation suite for network slicing in 5G networks, https://github.com/cerob/slicesim, 2021.