


Article

Effects of Transport Network Slicing on 5G Applications

Yi-Bing Lin [†], Chien-Chao Tseng [†] and Ming-Hung Wang ^{*,†} 

Department of Computer Science, National Yang Ming Chiao Tung University (NYCU), Hsinchu 300, Taiwan; liny@nctu.edu.tw (Y.-B.L.); cctsen@cs.nctu.edu.tw (C.-C.T.)

* Correspondence: mhwang408@gmail.com

[†] National Chiao Tung University (NCTU) and National Yang-Ming University (NYMU).

Abstract: Network slicing is considered a key technology in enabling the underlying 5G mobile network infrastructure to meet diverse service requirements. In this article, we demonstrate how transport network slicing accommodates the various network service requirements of Massive IoT (MIoT), Critical IoT (CIoT), and Mobile Broadband (MBB) applications. Given that most of the research conducted previously to measure 5G network slicing is done through simulations, we utilized SimTalk, an IoT application traffic emulator, to emulate large amounts of realistic traffic patterns in order to study the effects of transport network slicing on IoT and MBB applications. Furthermore, we developed several MIoT, CIoT, and MBB applications that operate sustainably on several campuses and directed both real and emulated traffic into a Programming Protocol-Independent Packet Processors (P4)-based 5G testbed. We then examined the performance in terms of throughput, packet loss, and latency. Our study indicates that applications with different traffic characteristics need different corresponding Committed Information Rate (CIR) ratios. The CIR ratio is the CIR setting for a P4 meter in physical switch hardware over the aggregated data rate of applications of the same type. A low CIR ratio adversely affects the application's performance because P4 switches will dispatch application packets to the low-priority queue if the packet arrival rate exceeds the CIR setting for the same type of applications. In our testbed, both exemplar MBB applications required a CIR ratio of 140% to achieve, respectively, a near 100% throughput percentage with a 0.0035% loss rate and an approximate 100% throughput percentage with a 0.0017% loss rate. However, the exemplar CIoT and MIoT applications required a CIR ratio of 120% and 100%, respectively, to reach a 100% throughput percentage without any packet loss. With the proper CIR settings for the P4 meters, the proposed transport network slicing mechanism can enforce the committed rates and fulfill the latency and reliability requirements for 5G MIoT, CIoT, and MBB applications in both TCP and UDP.



Citation: Lin, Y.-B.; Tseng, C.-C.; Wang, M.-H. Effects of Transport Network Slicing on 5G Applications. *Future Internet* **2021**, *13*, 69. <https://doi.org/10.3390/fi13030069>

Academic Editor: Christos Tranoris

Received: 31 January 2021

Accepted: 8 March 2021

Published: 11 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: Fifth generation network (5G); Network Slicing; Internet of Things (IoT); Software-Defined Network (SDN); Massive IoT (MIoT); Critical IoT (CIoT); Programming Protocol-Independent Packet Processors (P4)

1. Introduction

To meet diverse usage scenarios and service requirements in 5G, the 3rd Generation Partnership Project (3GPP) introduces network slicing to partition a physical network infrastructure into logical networks and assign dedicated hardware resources per slice tailored to a specific service. As defined in [1], a network slice is a logical network that can provide particular network capabilities and characteristics to respective traffic types, allowing for more efficient distribution and utilization of network resources and more flexibility for service deployment. In this article, we demonstrate how the proposed transport network slicing accommodates the various network service requirements of Massive Internet of Things (MIoT), Critical IoT (CIoT), and Mobile Broadband (MBB) applications.

Unlike the prior monolithic 4G core network, 5G adopts a Service-Based Architecture (SBA) and decouples the core network into several fine-grained and modular network

functions (NFs) that can be easily chained together to form a loosely-coupled control plane. Figure 1 illustrates a simplified 5G system, where the User Equipment (UE; Figure 1 (1)) accesses the Data Network (DN; Figure 1 (2)) through the Radio Access Network (RAN; Figure 1 (3)) and the User Plane Function (UPF; Figure 1 (4)). The Session Management Function (SMF) creates, updates, and removes the Protocol Data Unit (PDU) sessions, manages the session contexts with the UPF, and interacts with it through the N4 interface implemented in the Packet Forwarding Control Protocol (PFCP). The Access and Mobility Management Function (AMF) handles connection and mobility management tasks for the UE through the N2 interface. The AMF conducts the UE authentication process by interacting with the Authentication Server Function (AUSF) to authenticate UEs. The AUSF accesses the Unified Data Management (UDM) for UE registration and other functions. The Policy Control Function (PCF) specifies the control plane functions' policies, including network slicing, roaming, and mobility management. It also supports other UE access policies to be exercised by the UDM. The Network Slice Selection Function (NSSF) selects the AMF sets and the network slice instances to serve the UEs. The NF Repository Function (NRF) maintains the necessary functions for service registration and recovery of core network functions.

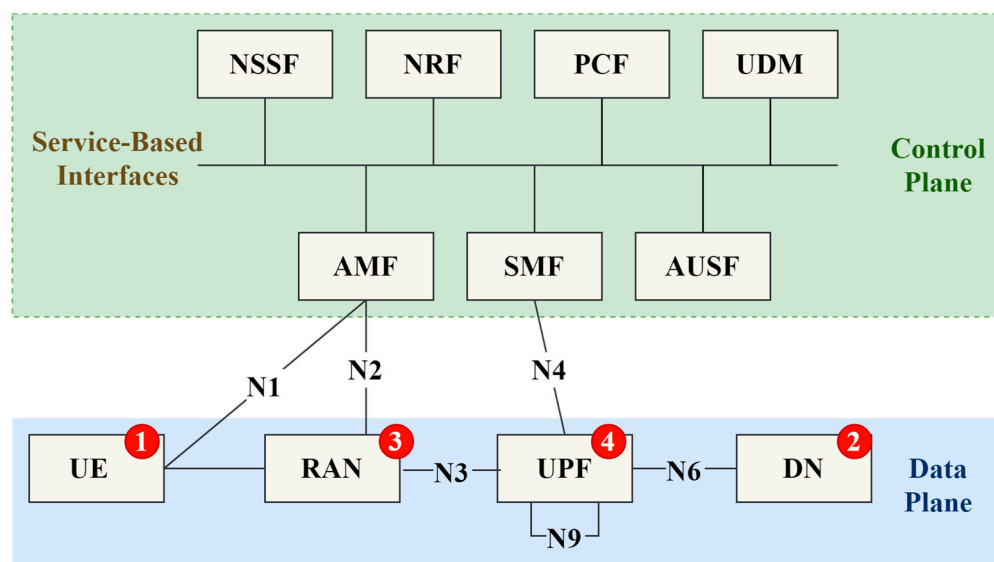


Figure 1. 5G Service-Based Architecture (SBA). The control-plane network functions that evolved from the 4G core network are decomposed into several microservices that interact with each other through Service-based interfaces.

An End-to-End (E2E) network slice in 5G mobile networks spans the RAN, the transport network, and the core network. The study in [2] describes an E2E network slice orchestration platform for the 4G Evolved Packet Core (EPC), where the base stations query the E2E slicer to select the Mobility Management Entity (MME) upon UE attachment. The research in [3] describes how 5G E2E network slicing is orchestrated and managed, and proposes a RAN slicing model for 5G services. The work in [4] presents an E2E network slicing testbed based on open-source tools capable of supporting slice provision dynamicity, real-time monitoring of virtual machines, and Virtual Network Function (VNF) instantiation. In [5], the authors introduce a Software-Defined Networking (SDN)-enabled 5G experimental platform that focuses on the RAN and the core network. The study [6] also focuses on RAN slicing and proposes an SDN-based scheduler to enable the dynamic slicing of radio resources among IoT and enhanced Mobile Broadband (eMBB) applications. The proposed solution emulates a Cloud Radio Access Network (C-RAN) scenario by using the OpenAirInterface (OAI) platform [7] and the FlexRAN SDN Controller [8]; then measures the efficiency of the proposed solution through emulations. In the study [9], the

authors introduce a mathematical model based on combinatorial designs for E2E network slicing. In the simulation results, the optimal strategy for the Network Slice Management Function (NSMF) can not only maximize the utilization of the network components but also decrease the average delay time of slice instantiation, configuration, and activation.

Transport networks provide connectivity between disaggregated RAN components and the core network. The RAN disaggregation and emerging service requirements drive the need for a flexible architecture for transport networks. Thus, the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) introduces a reference architecture for SDN control of transport networks [10]. With SDN's flexibility, network slicing in transport networks can differentiate or isolate traffic flows, dedicate resources to a network slice instance, and enforce slice-specific requirements. The study [11] leverages SDN technologies to learn a global view of network resource allocation and then calculates forwarding paths accordingly. Nonetheless, it comes with significant control plane overhead for scheduling while network complexity grows, and may fail to optimize resource utilization with path allocation.

Researchers [12–14] utilize data plane programmability to develop Quality of Service (QoS) management solutions using Programming Protocol-Independent Packet Processors (P4) switches. The study [12] proposes a customized P4 pipeline for strict bandwidth limitation and UDP stream guarantees. The earlier study [13] found that the P4 meters in commodity P4 switches can only reach a 10% target rate for TCP streams. As such, they proposed TCP-friendly meters that can achieve up to 85% in target rate. The work in [14] utilizes the P4 NetFPGA reference implementation to develop a QoS-aware data plane for slicing that classifies traffic flows and schedules them to multi-level priority queues, but the study only considers the effects of network slicing on media-rich applications.

Most of the studies mentioned above focus on the orchestration and management of either E2E or RAN network slicing while barely addressing transport network slicing. Moreover, none of these studies measure the effects of network slicing on a broad range of applications. By contrast, this article utilizes real applications and their emulated counterparts to generate traffic flows and conduct empirical evaluations in a hardware testbed, which is more realistic than the simulation approach.

To investigate 5G network slicing performances, we have deployed several smart applications to provide traffic flows to an SDN-based 5G testbed at National Chiao Tung University (NCTU). However, it is challenging to generate actual IoT application traffic on a large scale, which demands massive deployment of IoT devices of various types. As such, we utilize SimTalk, an IoT application traffic emulator based on the IoT application development platform IoTalkTM [15], to accommodate actual IoT applications and emulate their traffic patterns to create large-scale traffic. We then examine how the proposed transport network slicing mechanism affects the performance of these applications using the NCTU testbed (described in Section 2).

This study's major contribution lies in evaluating the effects of transport network slicing on 5G applications by utilizing the traffic flows from both actual applications and their emulated counterparts. Our study indicates that applications with different traffic characteristics need different Committed Information Rate (CIR) ratios. The CIR ratio is the CIR setting for a P4 meter in physical switch hardware over the aggregated data rate of applications of the same type. A low CIR ratio will cause an application to experience performance degradation because the P4 switches will dispatch application packets to the low-priority queue if the packet arrival rate is higher than the CIR setting for the applications. In general, applications with TCP or UDP elephant flows would require a higher CIR ratio than applications with TCP mice flows. With the proper CIR settings for the P4 meters, the proposed transport network slicing mechanism can enforce the committed rates and fulfill the latency and reliability requirements for 5G MIoT, CIoT, and MBB applications in both TCP and UDP.

This article is organized as follows. Section 2 introduces the main components in the NCTU testbed. We then describe the 5G applications for network slicing in Section 3.

Section 4 describes a P4-based slice and bandwidth management mechanism. Section 5 provides an overview of the SimTalk architecture. Section 6 investigates how transport network slicing improves the performance of 5G applications. Finally, we summarize the findings in our study and point out future work directions in Section 7.

2. The NCTU Testbed

At NCTU, we have developed a 5G testbed based on P4 switches and free5GC™ [16], an open-source solution for 5G core networks. Figure 2 shows the architecture of the NCTU testbed, where the UE (Figure 2 (1)) accesses the IoTalk services (Figure 2 (2)) through the RAN (Figure 2 (3)), and the UPF implemented in a Programming Protocol-Independent Packet Processors (P4) switch (Figure 2 (4)). This implementation decouples the UPF into a PFCP agent on the Software-Defined Network (SDN) controller (Figure 2 (5)) as the UPF control plane and a UPF data plane in the P4 switch. Besides, a P4-based transport network (Figure 2 (6)) with one or more P4 switches connects the RAN to the P4 UPF and other free5GC network functions (Figure 2 (7)). The P4 transport network is where we exercise the network slicing on various applications.

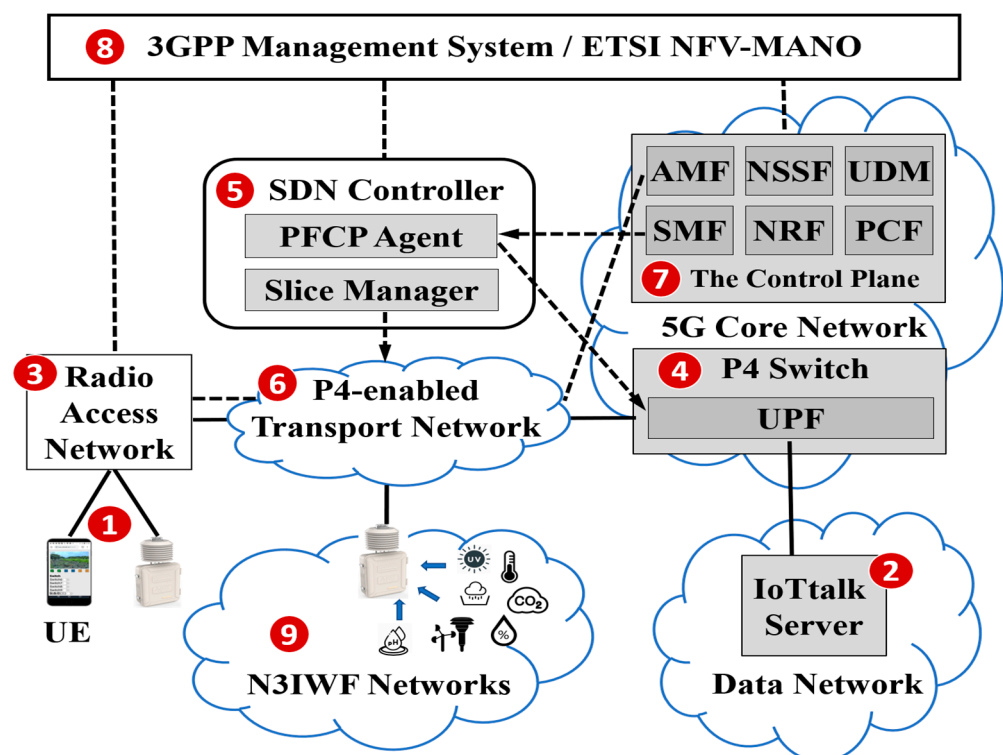


Figure 2. An SDN-based 5G architecture using P4 switches on the National Chiao Tung University (NCTU) campus. All IoT devices and UEs (1) access an IoTalk server (2) in the DN through either the RAN (3) or the Non-3GPP Inter-Working Function (N3IWF) networks (9). All application traffic is forwarded through a P4-enabled transport network (6) managed by an SDN controller (5) as well as a P4 UPF (4) managed by the 5G control plane (7). A 3GPP management system (8) is a management plane that carries out an E2E network slice through (3), (5), and (7).

The P4 switches employed in our testbed include the Edgecore Wedge 100BF-65X (65 × 100 Gb/s), Edgecore Wedge 100BF-32X (32 × 100 Gb/s), Inventec D5264 (64 × 100 Gb/s), and Inventec D10056 (48 × 25 Gb/s + 8 × 100 Gb/s), and are controlled by the SDN controller. Aside from the control and user planes, 3GPP also defines the network management architecture for the 5G system [17]. The 3GPP Management System can interact with the ETSI Network Function Virtualization MANO (NFV-MANO) specification for E2E

network slicing; that is, the 3GPP Management System and ETSI NFV-MANO (Figure 2 (8)) manages resources through the RAN, the SDN controller, and the 5G control plane.

In this testbed, a UE communicates with the AMF through the path (1)-(3)-(6)-(7). The AMF contacts the NSSF to identify the network slice of the UE and selects an active SMF to delegate the management of the UE's PDU session to said SMF. The SMF then establishes a PDU session on an appropriate UPF. After completing this establishment, the UE can access network services through the given UPF. In our testbed, the SMF in the control plane controls the UPF in the P4 switch through the PFCP agent in the SDN controller, which is responsible for issuing the corresponding SDN instructions to the P4 UPF; the path for this process is (7)-(5)-(4).

All application traffic is directed to the NCTU testbed and serviced by corresponding network slices managed by an NFV Orchestrator (NFVO) in the 3GPP Management System. We would also like to point out that most applications are directly connected to the NCTU testbed through the N3IWF networks (Figure 2 (9)) without passing through the expensive 5G new radio. Thus, this article describes the 5G applications that take advantage of SDN, NFV, and network slicing features in the transport and the core networks, but not the 5G new radio.

3. IoTtalk™ Applications for 5G Use Cases

Compared with 4G, the 5G core network has advanced significantly to support new service demands. Network services can be classified into three categories—the enhanced Mobile Broadband (eMBB) category targets high bandwidth access, the Ultra-Reliable and Low Latency Connection (URLLC) category targets reliable and latency-sensitive access, and the massive Machine Type Communications (mMTC) category targets machine-type communications in massive quantities [18,19].

In the NCTU testbed, we direct MBB applications that operate daily and mMTC applications to the 5G core network. We do not direct URLLC traffic to the current stage of the NCTU testbed because URLLC applications are mission-critical, requiring an E2E target latency of 1 ms and 99.999 percent reliability. One such application is autonomous driving [20], in which information is shared among the vehicles and the surrounding roadside environment in real-time. Information-sharing between neighboring vehicles would be achieved through vehicle-to-vehicle communications without being passed to the 5G core network. In contrast, information-sharing between a vehicle and roadside environment would occur through vehicle-to-infrastructure communications. 5G architecture adopts Multi-access Edge Computing (MEC) to assist vehicle-to-infrastructure communications, and such traffic does not need to go through the 5G core network either.

In 5G services, eMBB is supposed to provide broadband applications with moderate latency improvements on 4G. In the NCTU testbed, we developed emerging AR media applications with EPSON BT-30C smart glasses, connecting a pair of AR glasses (Figure 3a (1)) to the NCTU testbed through a smartphone (Figure 3a (2)). We also developed several monitoring tools that support video streaming, exemplified by our invention of a lobster farm monitoring system in Tainan City. Such eMBB video applications require the latency to be less than 50 ms, the user data rate to be 2 Mb/s, and the reliability 99.99%. Therefore, we apply the video streams to emulate eMBB traffic flows and examine network slicing effects on eMBB applications.

We have already employed many mMTC applications using 3GPP R13 or R14 low-power wide-area technology (i.e., NB-IoT), as well as LoRA and Sigfox, on the NCTU campus. We could have connected the sensors/actuators of these applications to our 5G testbed through the N3IWF networks. These applications are also referred to as MIIoT applications because most of the mMTC data periodically produced by their sensors are not critical. MIIoT applications typically have a packet size of 1 KB while requiring a latency of less than 1000 ms and reliability of 99.99%. Although the NCTU campus does not support URLLC, we do provide CIIoT applications, such as smart switches with a latency of less than 20 ms and a 99.999% reliability.



Figure 3. The MBB, the MIoT, and the CIoT applications based on IoTtalk™. (a) An AR application. A pair of smart glasses (1) connected to the NCTU testbed through a smartphone (2). (b) A NB-IoT-based application for on-campus parking at NCTU. (c) A dashboard for agriculture sensors. By clicking on its icon, the user can access the time-series graph of a sensor. (d) The piglet crush video monitor consists of a screen (3) displaying live streaming of farrowing pens, a dashboard (4) showing temperature and microphone status, and a control board (5) with toggle switches controlling the actuators in the pens.

This section will introduce several broadband and IoT applications that have operated sustainably on the NCTU campus for more than three years (see Appendix A). All applications were developed based on IoTtalk™ and the related tools are listed in Appendix ???. In these applications, the IoTtalk™ devices are UEs, sensors, and actuators connected to RAN or N3IWF networks and the corresponding network services are hosted on the IoTtalk™ server located at the data network. These applications themselves are IoT application platforms called “X-Talk” platforms. The information sources of the “X-Talk” platforms are given in Appendices A.2–A.6, and the details are elaborated below.

ElevatorTalk (Appendix A.2) is an elevator development and management system. This system modularizes elevator software components into IoT devices connected to the IoTtalk™ server to facilitate the development of flexible and scalable elevator scheduling algorithms. All messages exchanged between the devices and the server are critical, therefore making ElevatorTalk a CIoT application.

AgriTalk™ (Appendix A.3) is a precision farming system for soil cultivation involving MIoT (e.g., sensors for temperature, humidity, and pH), CIoT (e.g., actuators for pesticide

sprayers, fertilizer drippers, etc.), and MBB (e.g., cameras monitoring the farm fields) applications. Figure 3c exhibits the system dashboard for agricultural sensors. The user can access the time-series graph of a sensor by simply clicking on its icon.

CampusTalk (Appendix A.4) is a system that integrates a collection of MIoT applications (e.g., washing machines, dryers, parking lots, and bus scheduling), CIoT applications (e.g., smart switches, sprinklers, drippers, and lights), and MBB applications (e.g., security cameras) on campus. Note that the application for on-campus parking at NCTU uses the wireless technology NB-IoT, which is an MIoT application. (Figure 3b).

IoAtalk™ (Internet of Arts; Appendix A.5) is a series of interactive art applications that allow audiences to interact with the artworks in the form of CIoT. If the artworks are animations, the actuators are MBB.

HusbandryTalk (Appendix A.6) includes a series of applications for aquariums, bat farms, pig farms, lobster farms, and more. Such applications involve MIoT (e.g., sensors for temperature, dissolved oxygen, or pH), CIoT (e.g., actuators for heaters, fans, and air pumps), and MBB (e.g., cameras monitoring the aquariums).

In the IoTtalk™ system, the displays for MIoT, CIoT, and MBB can be merged into a single web-based window. For example, Figure 3d exhibits the web-based display for the application PigTalk. The video screen (3) in Figure 3d is an MBB application that allows the hog farmer to monitor the farrowing pens in real-time. The dashboard (4) in Figure 3d displays the temperature and microphone status; the former is an MIoT application, while the latter is a CIoT application collecting audio data from the pigs in the farrowing pens. The control board (5) in Figure 3d is a CIoT application that allows the hog farmer to turn on or off the actuators in the pens, such as those responsible for floor vibrations, air blasts, sprinklers, and electrodes. Many piglets die in farrowing houses because they are crushed when sows roll over or lie down. If the sow does not stand up within one minute when crushing occurs, the piglet will die. To reduce the piglet mortality rate, PigTalk analyzes the raw audio data collected from the microphones. If the analysis indicates that a piglet is being crushed, IoTtalk™ will automatically trigger actuators responsible for functions such as floor vibration and water drop to force the sows to stand up. The control board also allows the hog farmer to immediately turn off the actuators in the event of a false alarm. We note that the video image obtained by the cameras can be inputted into an AI model (e.g., Yolo) to extract the image bounding box data, which can then be used together with other sensors to perform sensor fusion for behavior analysis of the sows and piglets.

At NCTU, all smart applications are managed by MapTalk (see Appendix ??), an Integrated Operations Center (IOC) application that allows users to view all IoTtalk™ applications through Google map. Similar to the way a traffic control center would employ an IOC, MapTalk serves as the IOC for all NCTU on-campus applications. While a traditional IOC is displayed on a huge screen in the control room, MapTalk is designed to be viewed through a smartphone, as illustrated in Figure 4. For example, when we click the “pH” button (Figure 4a (1)), the application will display the pH values of the cesspools of the Science Parks in Taiwan. When we click the “Farm” button (Figure 4b (2)), the locations of the NCTU farms will be shown on the map. Clicking on a farm icon (Figure 4b (3)) will call up either the dashboard (Figure 3c) or an integrated window of the video screen, the dashboard, and the control board (Figure 3d).

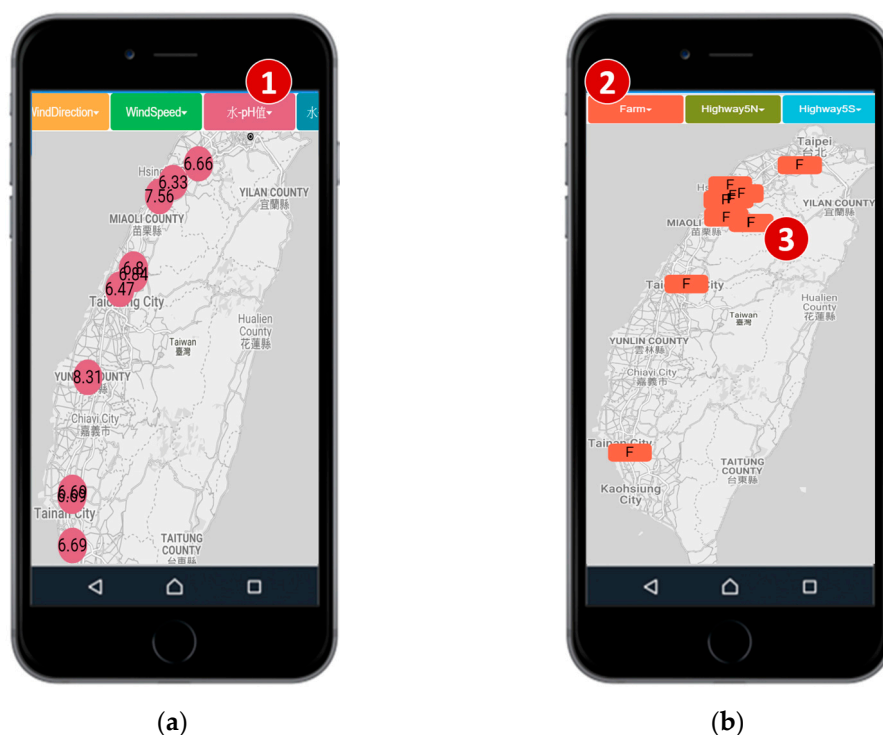


Figure 4. The MapTalk interface allows users to observe all sensor data through Google Maps. For example, users can view the pH values of cesspools or the locations of the NCTU farms by merely clicking the “pH” button (1) or the “Farm” button (2). (a) The pH values of cesspools. (b) The NCTU farms.

4. The SimTalk Architecture

The purpose of SimTalk [21], an extended feature based on the framework of IoTtalk™, is to develop a time-driven simulation that can automatically translate simulation codes into software codes running on physical IoT devices and vice versa. SimTalk is available in both simulation and emulation modes. In the simulation mode, the developer should specify the overall packet interval contributed by a group of devices; that is, a single probability distribution drives the total traffic generated by n devices (e.g., fixed, exponential, or gamma). In the emulation mode, the developer should specify the packet interval for a single device as those of its real counterparts; that is, the total traffic generated by n devices with the same packet interval is driven by n probability distributions, one per device. Therefore, SimTalk emulates the application not only in behavior but also in traffic patterns.

The IoTtalk application consists of various IoT devices and an IoTtalk service that manipulates these devices through wired or wireless technologies, including WiFi, LTE, NB-IoT, RoLa, and more. To facilitate the development of IoT applications, the IoTtalk server provides an IoTtalk GUI (Figure 5g) and a SimTalk GUI (Figure 5k) that allow the developers to create configurations of IoTtalk services and associated device models. Without actual coding, the IoTtalk server can provision the IoTtalk application from those configurations; that is, the IoTtalk server produces software components for IoT devices from the device models’ configurations, deploys the produced software components on the physical IoT devices or device simulators/emulators, creates instances of the IoTtalk services from the IoTtalk service’s configurations, and runs all active IoTtalk service instances accordingly.

In the IoTtalk platform, IoT devices with the same properties are modeled by a device model—an abstraction of the physical device. Two forms of device model, the input and output device models, accommodate one or more Input Device Features (IDFs) and Output Device Features (ODFs), respectively. For example, an input device model for a weather station may include two IDFs— one for the humidity sensor and the other for

the luminance sensor; an output device model for a farming actuator may consist of an ODF for the sprayer actuator. According to the device models' configuration, the AutoGen Subsystem (Figure 5i) automatically derives the software components of IoT devices, i.e., the Sensor and Actuator Application (SA) and the Device Application (DA). The SA implements device functionalities such as the PM2.5 algorithm or the light intensity and color circuit software. The DA processes communications with the IoTalk server. These can either be automatically generated (AG) (Figure 5a,b) or manually created (Figure 5c). The SA and DA can be bound to a physical device or a device emulator/simulator. An input device's emulator/simulator is a traffic generator that implements the device behavior through two probability distributions, one for the value generation and the other for the packet interval generation. The details for the output device's emulator/simulator are omitted since this article focuses on the traffic generation feature.

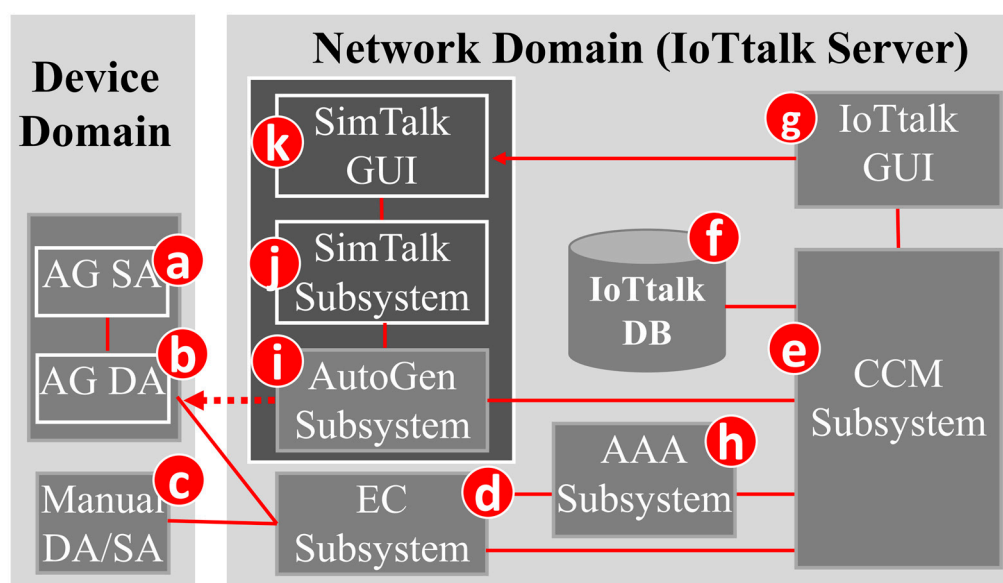


Figure 5. The architecture of IoTalk platform with SimTalk. The platform consists of components (d)–(i). Components (j) and (k) are SimTalk's components providing the simulation or emulation features, wherein (j) make use of IoTalk's component (i) to automatically derive the software components (a) and (b) for device emulators/simulators. Component (c) is manually created by the users. Component (d) executes instances of IoTalk services and interacts with components (a), (b), and (c) to carry out IoTalk applications.

The IoTalk service can be represented by a graph detailing the control and data dependencies among IoT devices. For example, a smart farm service may consist of a sequence of actions: (1) receiving luminance and humidity data from the sensors of the weather stations, (2) storing the received data in the IoTalk database (DB; Figure 5f) for data analysis and visualization, and (3) sending instructions to the relevant light and sprayer actuators under certain conditions. For the creation, configuration, and execution of the IoTalk services, the IoTalk server implements several subsystems, including the Creation, Configuration, and Management Subsystem (CCM), the Execution and Control Subsystem (EC), the SimTalk Subsystem, the AutoGen Subsystem, and the Authentication, Authorization, and Accounting (AAA) Subsystem (Figure 5h). The CCM Subsystem (Figure 5e) systematically creates and stores IoTalk service instances, along with the device models and their DFs, in the IoTalk DB. The EC Subsystem (Figure 5d) executes the IoTalk service instances, receives data from the IDFs, and sends instructions to the ODFs on behalf of the service. The EC Subsystem is also responsible for ensuring the processes above in alignment with the IoTalk service configuration.

In the SimTalk Subsystem (Figure 5j), an event handler receives instructions from the SimTalk GUI and the AutoGen Subsystem and executes simulator management procedures

corresponding to these instructions; this includes querying/saving parameters of device models and the creation of device simulator. The event handler stores the execution results in the SimTalk DB. In the AutoGen Subsystem, an event handler receives instructions from the SimTalk Subsystem and the CCM Subsystem and executes corresponding AutoGen management procedures to create or delete device simulators.

5. The Network Slicing and Bandwidth Management

To enforce 5G network slicing on the transport network, we implemented a bandwidth management mechanism [12] to regulate link usages. Figure 6 depicts its architecture, composed of a control plane running on the SDN controller and a data plane exercised in P4 switches.

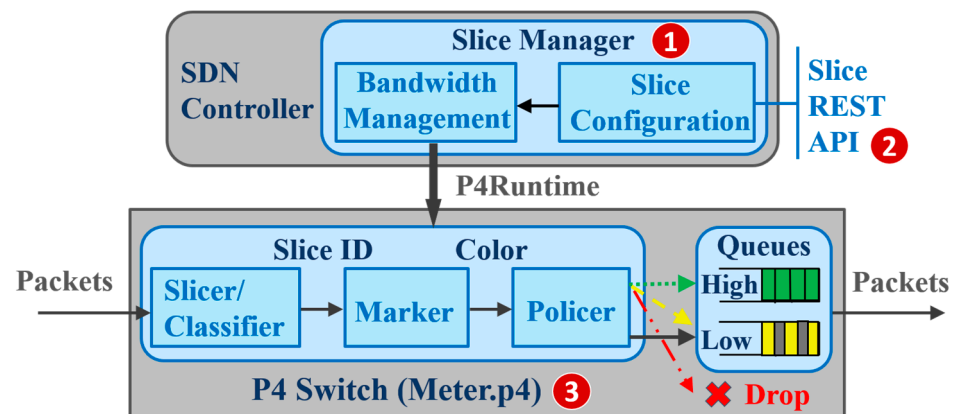


Figure 6. P4-enabled slice and bandwidth management mechanism. The Meter.p4 (3) is a customized data plane for strict bandwidth guarantee and rate-limiting. The Slice Manager (1) exposes a Slice REST API (2) for the Network Function Virtualization Orchestrator (NFVO) and manipulates flow rules in the data plane through the P4Runtime APIs.

Slice Manager (Figure 6 (1)), an ONOS [22] application, works as the control plane and exposes a Slice REST API (Figure 6 (2)) for slice configuration and bandwidth management. Slice Manager follows specifications given by the NFVO through the Slice REST API, turns the slice specifications into flow rules, and feeds the flow rules to the data plane. A slice could be a single flow identified by the five-tuple or a set of flows with the same IP destination or other granularities.

The data plane leverages the Two Rate Three Color Marker (trTCM) [23] of the P4 Meter for packet classification. The trTCM meters a data flow stream and marks packets of the flow according to four user-configurable parameters—the Peak Information Rate (PIR), the CIR, and their respective burst sizes. The PIR is the highest data rate allowed for the flow, while the CIR is the lowest flow rate guaranteed by the network. Specifically, trTCM marks an incoming flow packet as green, yellow, or red by comparing the metered flow rate with the PIR and CIR settings. The packet is marked red if the rate exceeds the PIR, green if the rate is lower than the CIR, and yellow if the measured rate is lower than the PIR but higher than the CIR. The bandwidth management mechanism guarantees delivery for green packets, delivers yellow packets with best-effort, and discards all red packets.

A P4 program called Meter.p4 (Figure 6 (3)) implements the above mechanism, performing packet classification, traffic policing, and traffic shaping. Meter.p4 defines a data plane pipeline of three stages—the Classifier/Slicer, the Marker, and the Policer, while Slice Manager is responsible for populating the flow rules on the P4 pipeline. The Classifier/Slicer stage identifies the slice of an incoming packet via the populated flow rules. It keeps the corresponding slice ID in a metadata field associated with the packet. The Marker stage uses the slice ID to select the corresponding meter from an array of trTCM meters. The meter then classifies packets into green, yellow, or red, based on the metered flow rate

and the PIR and CIR settings. The Policer stage dispatches green and yellow packets to high-priority and low-priority queues, respectively, and discards all red packets. This stage also dispatches packets of non-slicing flows to the low-priority queue. All packets in the high-priority queue are guaranteed with delivery, provided that the total bandwidth of all slicing flows does not exceed the link capacity. In contrast, the low-priority queue can only utilize the forwarding link to deliver yellow packets with best-effort when the high-priority queue is empty.

To investigate network slicing effects on IoT applications, we fed real traffic from many applications described in Section 3 into the NCTU testbed. Therefore, we have measured and obtained the traffic characteristics of these applications, and then emulated these applications accordingly through SimTalk. In the emulation mode of SimTalk, real applications and their emulated counterparts can coexist in the experiments, making it much more realistic than the simulation approach.

6. Impact of Network Slicing on 5G Applications

This section elaborates on the applications' traffic patterns and the experiments conducted to investigate the impact of transport network slicing. The input parameters we use for SimTalk to generate the emulated traffic of the same type include the packet size, the sending period, and the number of devices. The packet size and sending period we choose to generate the traffic for an application are those we observed from the real-world counterpart application running on our testbed. However, the number of devices we apply for the same application type is the maximum number of devices the server hardware can support. We define a CIR ratio as the CIR setting for a P4 meter in physical switch hardware divided by the aggregated data rate of applications of the same type, representing the relative ratio of the committed rate to the application traffic input rate. By varying the CIR ratios, we can study the effects of transport network slicing on an application under different bandwidth guarantees. Note that we focus on the effectiveness of bandwidth guarantee instead of rate-limiting and set the PIR parameter for a P4 meter to a constant value of line capacity in all experiments. The details of exemplar CIoT, MIoT, and MBB applications used in the experiments are elaborated below.

For the CIoT applications, we investigated the behavior of smart switches in the NCTU smart dorms. The raw sensor data of an IoT packet sent from the smart switch to the IoTtalk server is 1 byte. Encapsulated by the JSON format, this raw data gains 14 additional bytes, which comprises the payload of the IoTtalk HTTP message. The HTTP header includes variable-length fields, such as the device ID in URL, and the content-length field, taking up about 275 to 277 bytes in total. Consequently, the size of the whole IoTtalk HTTP packet ranges from 290 bytes to 292 bytes. Our experiments include 300 smart switches (called MorSocket; see Appendix A.4). According to our observation of the traffic patterns of the NCTU campus applications, SimTalk emulates various types of traffic that mix and form a Poisson process; the expected inter-arrival time can range from 0.01 s to 10 s. Each smart switch emulated by SimTalk has a packet-sending period of 0.2 s in the mean value.

For the MIoT applications, we use MapTalk to produce sensor data (e.g., temperature, humidity, and CO₂), indoor positioning data for sensor fusion, and map coordinates; the raw data size in total is 104 bytes. The IoTtalk HTTP message size amounts to 394 to 396 bytes. 10,000 MapTalk sensors are involved in our experiment, each producing a sample every 10 s.

The two MBB applications, namely sensor fusion and PigTalk, are elephant flows in our testbed. For sensor fusion, we use Yolo to analyze video frames from a set of cameras and output the image bounding box data, combined with other sensory data to perform sensor fusion. The sensor fusion client collects and packs a batch of image bounding boxes into a payload formatted by 520 KB-long JSON, and the IoTtalk HTTP header adds an extra 293 to 294 bytes. Our experiments include 104 sensor fusion clients, and the sending period for each client is a fixed rate of 0.1 s. For PigTalk, the piglet crush video monitor receives and displays live streaming of farrowing pens and uses 1.4 KB-long Real-time Transport Protocol (RTP) packets over UDP for video streaming.

In the uplink traffic experiments, the IoTtalk HTTP messages are delivered along the path (9)-(6)-(4)-(2) in Figure 2. The link capacity of (6) and (4) is 10 Gb/s. There are three sources for application traffic, the first being 300 smart switches that send traffic at a rate of 11.312 Mb/s. Additionally, 104 sensor fusion clients generate traffic with a sending rate of 4609.896 Mb/s, while 10,000 MapTalk sensors generate traffic at an average load of 8.296 Mb/s. We also include the background UDP traffic generated from the iPerf [24] with a data rate of up to 10 Gb/s.

In the downlink traffic experiments, a total of 10 PigTalk video streams travel along the path (2)-(4)-(6)-(9) in Figure 2. This MBB application results in a total sending rate of 2.632 Gb/s. The background UDP traffic is generated from the iPerf with a data rate of up to 10 Gb/s.

Figure 7 illustrates how network slicing improves the three measures of throughput, loss rate, and latency for application traffic flows. The graphs (a), (c), and (e) evaluate how application traffic flows, both baseline flows (namely non-slicing flows) and slicing flows, are affected by various background traffic volumes. On the other hand, (b), (d), and (f) illustrate how the output measures of the application traffic flows are affected by various CIR ratios when network links are saturated with 10 Gb/s background traffic.

In Figure 7, the dashed curves in graphs (a), (c), and (e) demonstrate how baseline flows are affected by various background traffic volumes, and the solid curves show how network slicing with a 100% CIR ratio improves traffic performance.

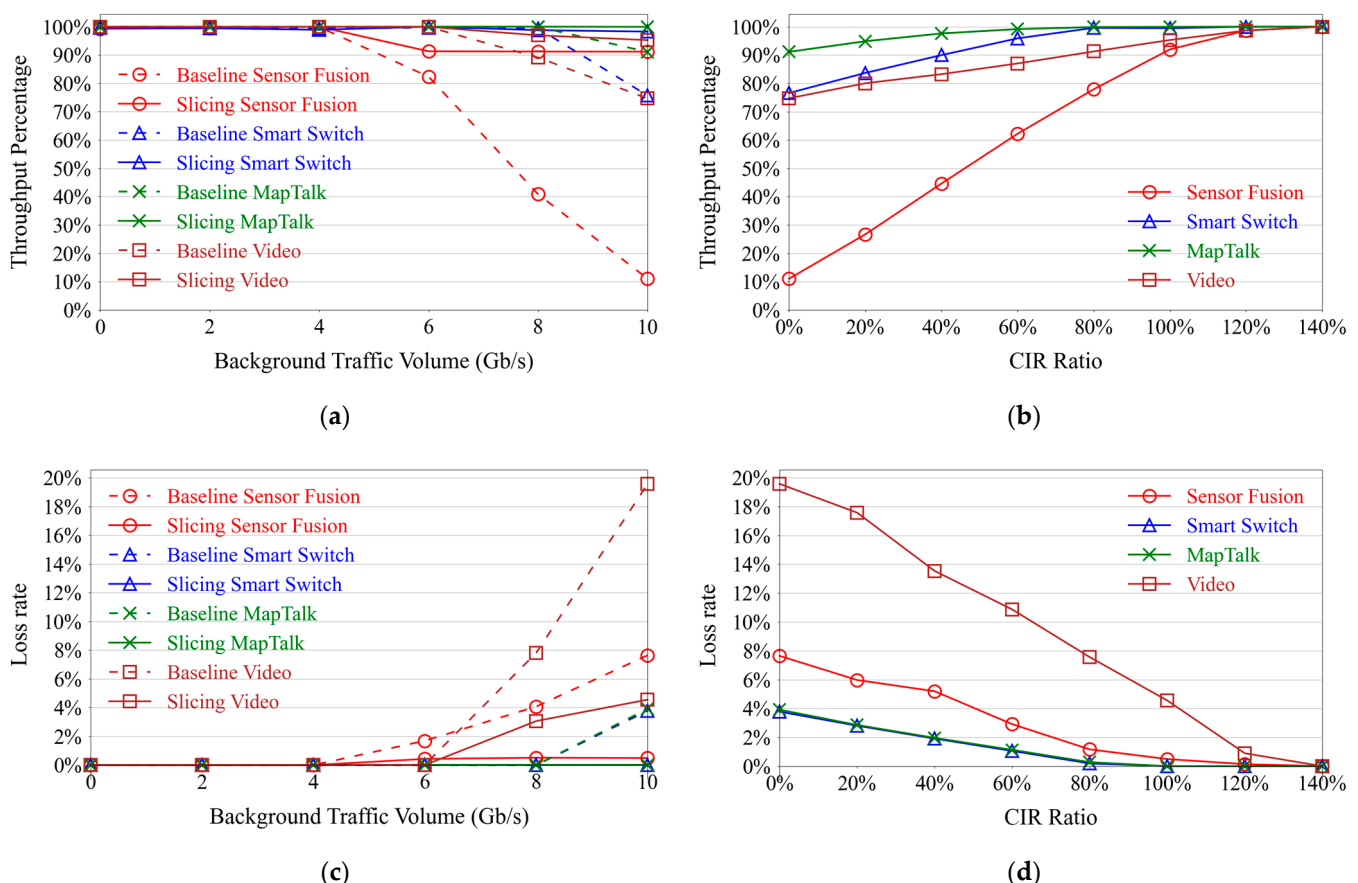


Figure 7. Cont.

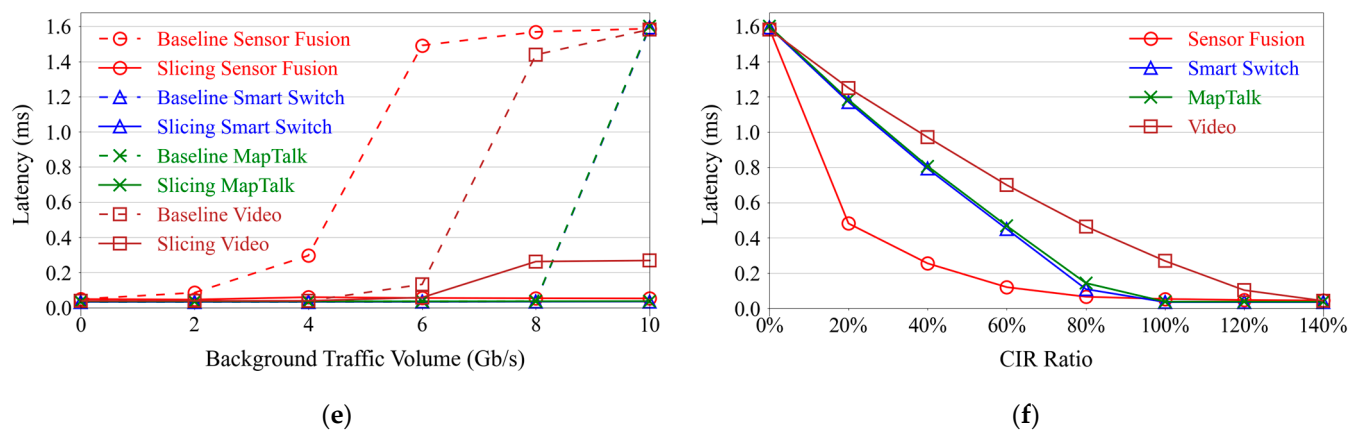


Figure 7. Effects of network slicing on applications' output measures of throughput, loss rate, and latency. Dashed curves represent the baseline applications' output measures; solid curves show the slicing applications' output measures. In graphs (a), (c), and (e), a slice with a CIR ratio of 100% is created and configured for each slicing application. Note that, in graphs (b), (d), and (f), the dots at the CIR ratio of 0% represent the output measures of baseline applications. In contrast, the dots at the positive CIR ratios represent the output measures of slicing applications. (a) Throughput percentages at different background traffic volumes. (b) Throughput percentages at different CIR ratios with the background traffic volume of 10 Gb/s. (c) Loss rates at different background traffic volumes. (d) Loss rates at different CIR ratios with the background traffic volume of 10 Gb/s. (e) One-way Latencies at different background traffic volumes. (f) One-way latencies at different CIR ratios with the background traffic volume of 10 Gb/s.

6.1. The Baseline/Non-Slicing Applications

In graphs (a), (c), and (e), all baseline applications experience significant performance degradation when the total demand is higher than the link capacity. When the volume of background traffic increases up to 6 Gb/s, the total demand exceeds the link capacity. In this case, an insufficiency of available bandwidth results in building up the low-priority queue, growing latencies, and packet loss occurrences. The reason is that the P4 switches forward all packets from baseline applications with the best effort and dispatch them to the low-priority queue, where these packets contend with the background traffic for the remaining available bandwidth. In graph (c), the baseline sensor fusion observed a loss rate of 0% with less than 6 Gb/s background traffic volumes; a series of positive loss rates of 1.69%, 4.08%, and 7.64% can be observed respectively when background traffic volumes increase to 6 Gb/s, 8 Gb/s, and 10 Gb/s. From graphs (a) and (c), we observed that the measured throughput for baseline sensor fusion is adversely affected by the packet loss, deteriorating once the loss rate rises above 0%. The reason is that when a packet loss occurs, the sender will receive multiple duplicate acknowledgments (ACKs) or encounter a retransmission timeout (RTO) for the loss; TCP congestion control will take both instances as an indication of network congestion and reduce the sending rate. The baseline smart switch and MapTalk have the same phenomenon because of the same reason. Nevertheless, the baseline UDP video keeps sending packets, regardless of congestion. Therefore, it encounters severe packet losses, and the packet losses account for reducing throughput for the video with UDP.

Graphs (c) and (e) demonstrate that latency is an advanced indicator for congestion. The steepest part of the dashed curve for baseline sensor fusion in graph (e) occurs before a positive loss rate occurs, as seen in graph (c). This result indicates that the low-priority queue builds up rapidly when the total demand for available bandwidth is about to reach the link capacity and stays in a near-overflow level when the link is saturated.

The dashed curves in graphs (a), (c), and (e) reveal that network congestion has a more significant impact on the baseline sensor fusion than on the baseline smart switch and baseline MapTalk. Note that these connections of sensor fusion are long-lived TCP elephant flows, whereas those of the smart switch and MapTalk are short-lived TCP mice

flows. The diversity of traffic patterns between these applications makes the impact of network congestion on applications different.

From the above discussion, we conclude that the performance degradation in the baseline applications is due to network congestion, leading to a severe impact on the baseline applications with TCP or UDP elephant flows and a minor effect on the baseline applications with TCP mice flows.

6.2. The Slicing Applications

To investigate how network slicing improves applications' performance, we configured a slice with a CIR ratio of 100% for each application on the P4 switches. The solid curves in graphs (a), (c), and (e) represent the output measures of the slicing applications, which gain a significant improvement over those of the baseline applications. The solid curves in graphs (c) and (e) show that, when the link is saturated, the loss rate and latency of the slicing video with UDP increase proportionally to the background traffic volume. This phenomenon does not occur with other slicing applications with TCP because TCP congestion control reduces the sending rate when congestion occurs, limiting the number of packets dispatched to the low-priority queue in the P4 switches. However, the slicing video with UDP has more packets marked yellow dispatched to the low-priority queue. These packets contend for available bandwidth with the background traffic. Therefore, the average latencies of the video with UDP increase in proportion to background traffic.

The solid curves in graphs (a) and (c) show that a 100% CIR ratio still leads to about 1.21% to 1.73% throughput degradation with almost 0% loss rates for the slicing smart switch, about 9% throughput degradation with about 0.4% to 0.5% loss rates for the slicing sensor fusion, and about 3% to 5% throughput degradation with about 3% to 5% loss rates for the slicing video. By contrast, the same CIR ratio reaches a throughput percentage of 100% without any packet loss for the slicing MapTalk, indicating that applications with different traffic patterns may require different CIR ratios. Therefore, we further studied the network slicing effect on applications' output measures by varying the CIR ratios to verify this idea.

Graphs (b), (d), and (f) in Figure 7 demonstrate that increasing the CIR ratio can improve the applications' output measures when the total demand is larger than the link capacity. These figures also confirm that the smart switch, the sensor fusion, and the video require a CIR ratio above 100%. In graph (d), the CIR ratios above 100% considerably reduce the loss rates of the smart switch, the sensor fusion, and the video. For the smart switch, the loss rate drops to 0% at a 120% CIR ratio. For the sensor fusion, the loss rate drops to 0.14% at a 120% CIR ratio and 0.0035% at a 140% CIR ratio. As for the video, the loss rate drops to 0.9% at a 120% CIR ratio and 0.0017% at a 140% CIR ratio. Figure 7b indicates that a reduction in the CIR ratio causes a more significant impact on the sensor fusion than other applications. Figure 7d shows that, with the same CIR ratio, the video has a higher loss rate than other TCP applications because of TCP congestion control.

7. Conclusions

This article demonstrates that network slicing is an effective mechanism for seamlessly integrating 5G MBB, MIoT, and CIoT applications, making it the key technology to fulfilling the diverse requirements of 5G applications. We utilize SimTalk to emulate the traffic patterns of actual applications running on NCTU campus and direct large-scale emulated traffic into a P4-based testbed to study network slicing impact on these applications. The performances are measured in terms of throughput, packet loss, and latency. According to the findings in [13], the P4 meters in commodity P4 switches can reach only 10% of the target rate for TCP streams. In contrast, the proposed slice and bandwidth management mechanism can enforce the committed rates for designated application flows in both TCP and UDP effectively and meet the 5G latency and reliability requirements for MIoT, CIoT, and MBB applications.

The proper CIR ratio for an application depends on the traffic characteristics. When the total demand is higher than the link capacity, a CIR ratio much higher than 100% is necessary for the applications with TCP or UDP elephant flows. On the other hand, a 100% CIR ratio is sufficient for the applications with TCP mice flows. Notably, the sensor fusion needs a 140% CIR ratio to achieve an approximate 100% throughput percentage with a 0.0035% loss rate. The video requires a 140% CIR ratio to achieve an approximate 100% throughput percentage with a 0.0017% loss rate. The smart switch and the MapTalk demand a CIR ratio of 120% and 100%, respectively, to achieve a 100% throughput percentage without any packet loss.

We are developing the NCTU testbed based on free5GCTM and investigating the possibility of building Aether [25], the first open-source Enterprise 5G/LTE Edge-Cloud-as-a-Service platform developed by Open Networking Foundation (ONF). In the future, we will integrate the current NCTU testbed with the Aether Connected Edge (ACE) and develop a 5G MEC architecture supported by network slicing, where a slicing network for URLLC will operate in the NCTU testbed with other smart applications mentioned in Section 3. Also, we will direct all traffic through the 5G new radio and conduct further empirical experiments to investigate how network slicing can support URLLC, eMBB, CIoT, and MIoT applications. Last but not least, the effects of the four user-configurable parameters of trTCM meters on these applications with different traffic characteristics are also worthy of more study to provide a guideline for practical usage scenarios.

Author Contributions: Conceptualization, M.-H.W.; Software, M.-H.W.; Supervision, Y.-B.L. and C.-C.T.; Validation, M.-H.W.; Writing—original draft, Y.-B.L., C.-C.T. and M.-H.W.; Writing—review & editing, Y.-B.L., C.-C.T. and M.-H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Center for Open Intelligent Connectivity from The Featured Areas Research Center Program within the Framework of the Higher Education Sprout Project by the Ministry of Education in Taiwan; in part by the Ministry of Science and Technology under Grant 108-2221-E-009-047, Grant 107-2221-E-197-006-MY3, Grant 108-2321-B-197-003 and Grant 109-2221-E-009-077; and in part by the Ministry of Economic Affairs under Grant 107-EC-17-A-02-S5-007.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

This appendix provides the documentary sources of the X-Talk applications used in the NCTU 5G testbed.

Appendix A.1. IoTalkTM Tools

This subsection provides the references to describe how NB-IoT and infrared communications are integrated with IoTalkTM, and how IoT devices are connected to IoTalkTM device applications through Arduino. We then provide the reference for MapTalk (see also Figure 4).

- Y.-B. Lin et al. "NB-IoTalk: A Service Platform for Fast Development of NB-IoT Applications," IEEE Internet of Things Journal, Vol.6, Issue 1, pp.928-939, February 2019. pdf
- Y.-W. Lin et al. "IoTalk-RC: Sensors as Universal Remote Control for Aftermarket Home Appliances," IEEE Internet of Things Journal, Volume 4, Issue 4, Aug. 2017. pdf
- Y.-W. Lin et al. "ArduTalk: An Arduino Network Application Development Platform Based on IoTalk," IEEE Systems Journal, Vol.13, Issue 1, pp.468-476, March 2019. pdf
- Y.-B. Lin et al. "MapTalk: Mosaicking Physical Objects into the Cyber World. Cyber-Physical Systems," Volume 4, Issue 3, Pages 156-174, 2018. pdf

Appendix A.2. ElevatorTalk

This subsection provides the references for smart elevator systems controlled by IoTalk™.

- L.-D. Van et al. "Green Elevator Scheduling Based on IoT Communications." IEEE Access, Volume: 8, Issue:1, Page(s): 38404–38415, December 2020. (DOI:10.1109/ACCESS.2020.2975248) pdf
- L.-D. Van et al. "An Intelligent Elevator Development and Management System." Accepted and to appear in IEEE Systems Journal. pdf

Appendix A.3. AgriTalk™

This subsection provides the references for smart agriculture based on IoTalk™.

- W.-L. Chen et al. "AgriTalk: IoT for Precision Soil Farming of Turmeric Cultivation," IEEE Internet of Things Journal, Vol. 6, No. 3, pp. 5209–5223, June 2019. pdf
- L.-D. Van et al. "PlantTalk: A Smartphone-based Intelligent Hydroponic Plant Box," Sensors Journal, Vol. 19, No. 8, pp.1763, 2019. pdf

Appendix A.4. CampusTalk

This subsection provides the references for smart campus applications based on IoTalk™.

- Y.-B. Lin et al. "CampusTalk: IoT Devices and Their Interesting Features on Campus Applications." IEEE Access. Volume: 6, Issue:1, Page(s): 26036–26046, December 2018. pdf
- Y.-B. Lin et al. "DormTalk: Edge Computing for the Dormitory Applications on Campus," IET Networks, Vol. 8, No. 3, pp.179–186, June 2019. pdf
- Y.-B. Lin et al. "MorSocket: An Expandable IoT-based Smart Socket System," IEEE Access, Volume: 6, Page(s): 53123–53132, 2018. pdf
- Y.-B. Lin et al. "Low-Cost Four-Dimensional Experience Theater Using Home Appliances," IEEE Transactions on Multimedia, Vol.21, Issue 5, pp. 1161–1168, May 2019. pdf

Appendix A.5. IoAtalk™ (Internet of Arts)

This subsection provides the references for interactive art applications based on IoTalk™.

- W.-S. Lai et al. "FrameTalk: Human and Picture Frame Interaction through the IoT Technology," Mobile Networks and Applications Journal, First Online: 28 May 2019. pdf
- C.-Y. Hsiao et al. "Flower Sermon: An Interactive Visual Design Using IoTalk, Mobile Networks and Applications Journal," Vol. 24, No. 3, June 2019. pdf

Appendix A.6. HusbandryTalk

This subsection provides the references for animal husbandry based on IoTalk™.

- Y.-B. Lin et al. "FishTalk: An IoT-based Mini Aquarium System," IEEE Access, Vol.7, Issue 1, pp. 35457–35469, December 2019. pdf
- W.-E. Chen et al. "PigTalk: an AI-based IoT Platform for Piglet Crushing Mitigation," Accepted and to appear in IEEE Transactions on Industrial Electronics. pdf

References

1. 3rd Generation Partnership Project (3GPP). TS23.501 System Architecture for the 5G System (5GS), V16.4.0. Available online: https://www.3gpp.org/ftp//Specs/archive/23_series/23.501/23501-g40.zip (accessed on 15 February 2021).
2. Afolabi, I.; Taleb, T.; Frangoudis, P.A.; Bagaa, M.; Ksentini, A. Network slicing-based customization of 5G mobile services. *IEEE Netw.* **2019**, *33*, 134–141. [CrossRef]
3. Li, X.; Ni, R.; Chen, J.; Lyu, Y.; Rong, Z.; Du, R. End-to-end network slicing in radio access network, transport network, and core network domains. *IEEE Access* **2020**, *8*, 29525–29537. [CrossRef]
4. Esmaeily, A.; Kravlevska, K.; Gligoroski, D. A cloud-based SDN/NFV testbed for end-to-end network slicing in 4G/5G. In Proceedings of the 6th IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, 29 June–3 July 2020.
5. Ramantas, K.; Kartsakli, E.; Irazabal, M.; Antonopoulos, A.; Verikoukis, C. Implementation of an SDN-Enabled 5G experimental platform for core and radio access network support. In Proceedings of the Interactive Mobile Communication Technologies and Learning (IMCL), Thessaloniki, Greece, 30 November–1 December 2017.

6. Costanzo, S.; Fajjari, I.; Aitsaadi, N.; Langar, R. Dynamic network slicing for 5G IoT and eMBB services: A new design with prototype and implementation results. In Proceedings of the 3rd Cloudification of the Internet of Things (CIoT), Paris, France, 2–4 July 2018.
7. Nikaien, N. OpenAirInterface Simulator/Emulator. Available online: <http://www.openairinterface.org/> (accessed on 15 February 2021).
8. Foukas, X.; Nikaein, N.; Kassem, M.M.; Marina, M.K.; Kontovasilis, K. FlexRAN: A flexible and programmable platform for software-defined radio access networks. In Proceedings of the 12th International Conference on emerging Networking EXperiments and Technologies (CoNEXT), Irvine, CA, USA, 12–15 December 2016.
9. Gligoroski, D.; Kravlevska, K. Expanded combinatorial designs as tool to model network slicing in 5G. *IEEE Access* **2019**, *7*, 54879–54887. [\[CrossRef\]](#)
10. International Telecommunication Union Telecommunication Standardization Sector (ITU-T). G.7702 Architecture for SDN control of transport networks, V1.0. March 2018. Available online: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.7702-201803-1!!PDF-E&type=items (accessed on 15 February 2021).
11. Shu, Z.; Taleb, T. A Novel QoS Framework for Network Slicing in 5G and Beyond Networks Based on SDN and NFV. *IEEE Netw.* **2020**, *34*, 256–263. [\[CrossRef\]](#)
12. Chen, Y.W.; Yen, L.H.; Wang, W.C.; Chuang, C.A.; Liu, Y.S.; Tseng, C.C. P4-enabled bandwidth management. In Proceedings of the 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), Matsue, Japan, 18–20 September 2019.
13. Wang, S.-Y.; Hu, H.-W.; Lin, Y.-B. Design and Implementation of TCP-Friendly Meters in P4 Switches. *IEEE/ACM Trans. Netw.* **2020**, *28*, 1885–1898. [\[CrossRef\]](#)
14. Wang, Q.; Alcaraz-Calero, J.; Ricart-Sanchez, R.; Weiss, M.B.; Gavras, A.; Nikaein, N.; Vasilakos, X.; Giacomo, B.; Pietro, G.; Roddy, M.; et al. Enable advanced QoS-aware network slicing in 5G networks for slice-based media use cases. *IEEE Trans. Broadcasting* **2019**, *65*, 444–453. [\[CrossRef\]](#)
15. Lin, Y.B.; Lin, Y.W.; Huang, C.M.; Chih, C.Y.; Lin, P. IoTtalk: A management platform for reconfigurable sensor devices. *IEEE Internet Things J.* **2017**, *4*, 1552–1562. [\[CrossRef\]](#)
16. Tan, T.J.; Hu, W.T.; Weng, F.L.; Chang, H.C.; Chen, J.C.; Lin, Y.B. *Demo: Free5GC*; NIST: Gaithersburg, MD, USA, 2019.
17. 3rd Generation Partnership Project (3GPP). TS28.530 Management and Orchestration; Concepts, Use Cases and Requirements, V16.1.0. December 2019. Available online: https://www.3gpp.org/ftp//Specs/archive/28_series/28.530/28530-g10.zip (accessed on 15 February 2021).
18. Soos, G.; Janky, F.N.; Varga, P. Distinguishing 5G IoT use-cases through analyzing signaling traffic characteristics. In Proceedings of the 42nd International Conference on Telecommunications and Signal Processing (TSP), Budapest, Hungary, 1–3 July 2019.
19. 5G-TRANSFORMER Project Deliverable D5.3. Experimentation Results and Evaluation of Achievements in Terms of KPIs. June 2019. Available online: http://5g-transformer.eu/wp-content/uploads/2019/06/D5.3_Experimentation_results_and_evaluation_of_achievements_in_terms_of_KPIs.pdf (accessed on 15 February 2021).
20. Choy, J.L.C.; Wu, J.; Long, C.; Lin, Y.-B. Ubiquitous and low power vehicles speed monitoring for intelligent transport systems. *IEEE Sens. J.* **2020**, *20*, 5656–5665. [\[CrossRef\]](#)
21. Lin, Y.W.; Lin, Y.B.; Yen, T.H. SimTalk: Simulation of IoT applications. *Sensors* **2020**, *20*, 2563. [\[CrossRef\]](#) [\[PubMed\]](#)
22. Berde, P.; Gerola, M.; Hart, J.; Higuchi, Y.; Kobayashi, M.; Koide, T.; Lantz, B.; O'Connor, B.; Radoslavov, P.; Snow, W.; et al. ONOS: Towards an open, distributed SDN OS. In Proceedings of the third workshop on Hot topics in software defined networking (HotSDN), Chicago, IL, USA, 22 August 2014.
23. Heinanen, J.; Guerin, R. A Two Rate Three Color Marker. 1999. Available online: <https://tools.ietf.org/html/rfc2698> (accessed on 15 February 2021).
24. Iperf—The Ultimate Speed Test Tool for TCP, UDP, and SCTP. Available online: <https://iperf.fr/> (accessed on 15 February 2021).
25. Parikh, A. *Enabling a New Era of Smart Enterprises—Aether: Private 5G Connected Edge Platform*; Open Networking Foundation: Menlo Park, CA, USA, 2020.