Cost-efficient Method for Managing Network Slices in a Multi-service 5G Core Network

Takuya Shimojo, Malla Reddy Sama*, Ashiq Khan, and Shigeru Iwashina
NTT DOCOMO, INC., Japan *DOCOMO Comm. Labs, Germany
{takuya.shimojou.gt, khan, iwashina}@nttdocomo.com; sama@docomolab-euro.com

***Abstract*—“Network slicing” means creating logically isolated virtual networs (slices) running different network functions for different services. However, this technology imposes a loss of multiplexing gain available in monolithic networks as resources allocated to each slice become exclusive and isolated. This situation leads to resource wastage. In this paper, we propose a concept called “per-group slicing” and an algorithm for efficiently automating mechanisms for service grouping and slice creation/accommodation to improve multiplexing gain and resource-usage efficiency. Moreover, the concept is evaluated in consideration of future services discussed in the Next Generation Mobile Networks (NGMN) initiative. In addition, the loss in multiplexing gain imposed by network slicing technology was quantitatively evaluated on a testbed, and a means of balancing accommodation of diverse services and resource wastage when a network slice is used was devised.**

Keywords— Network Slicing; Mobile Core Networks; Network Functions Virtualization (NFV); Evolved Packet Core (EPC); Software Defined Networks (SDN)

I. INTRODUCTION

In the 5G era, a wide variety of objects, such as vehicles, houses, wearable devices, robots, and sensors, will be connected to mobile networks [1]. Categories of network services and operators’ business areas that can be expected through the 5G network are illustrated in Figure 1. Regions A and B represent the current mobile business based on cellular networks, which is mainly based on mobile- and smart-phone subscribers generating most of its revenue. Continued evolution of network equipment, such as servers and switches, will enlarge the scope of these mobile business and services. On the other hand, Region C represents future services based on the “Internet of Things” (IoT), which uses massive numbers of low-power terminals, while services in Region D, such as remote surgery, will have stringent performance requirements (such as low latency, high reliability, and a small number of terminals) and will be provided through future core networks. This trend suggests that mobile operators will be required to meet such requirements from third-party service providers like the car industry while maintaining operational costs at a reasonable level.

Currently, mobile networks deliver mainly voice and data services through the Evolved Packet Core (EPC) architecture. EPC is an all-IP network architecture that serves all services and different types of user equipment (UE), such as smartphones and M2M devices [2]. EPC is composed of a Mobility Management Entity (MME) node for mobility management, a Home Subscriber System (HSS) node for managing UE subscription information, a Serving Gateway (SGW) node as a mobility anchor point, and a Packet Data Gateway (PGW) node as a gateway between the mobile network and an external network, e.g., the Internet, as shown in Figure 2. An Evolved NodeB (eNB), which is the base station providing LTE radio access, is connected to the MME and SGW via S1-MME (control plane) and S1-U interfaces (data plane), respectively [2]. EPC node is not programmable via the network according to demands of end users and types of services in a cost-effective manner.

Aiming to satisfy a wide variety of requirements in the 5G era, different telco organizations are working on the above-mentioned issues. For instance, NGMN uses the concept of “network slices,” which establish a service-based end-to-end dedicated virtual network by using two techniques, namely, slice leveraging the “network functions virtualization” (NFV) and “software-defined networking” (SDN) [3-5]. This slice concept can be one of the key features of a 5G network; however, the resources of functional entities allocated to each slice are exclusive and isolated. So a “slice-per-service” architecture, which allocates slice resource to every service to guarantee performance requirements would lead to a loss of multiplexing gain. Additionally, operators have to create and operate a massive number of slices for different services, so operational costs would increase. An effective way to decrease such operational costs is to create slices, where each slice hosts a group of similar network requirements, thereby reducing the total number of slices. In this paper, a “per-group slicing” architecture and a service-slice-mapping algorithm (for mapping a service group to an appropriate slice) are first proposed. Then, preliminary results of a simulation using the algorithm in reference to the future service requirements are presented. Finally, the amount of resource wastage when using the algorithm was evaluated in comparison with that of a slice-per-service operation on our in-lab testbed. The evaluation results indicate that the proposed algorithm reduces operational costs and multiplexing gain loss and can accommodate various 5G services.

The rest of the paper is organized as follows. Related works are presented in Section II. Requirements for a slice-management architecture are stated in Section III. A slicing-based service-management architecture and a service-slice mapping algorithm are proposed in Sections IV and V, respectively. The proposed algorithm is evaluated in terms of quantitative multiplexing gain in Section VI. Finally, the results and conclusions of this study are a summarized and future works are described briefly in Section VII.

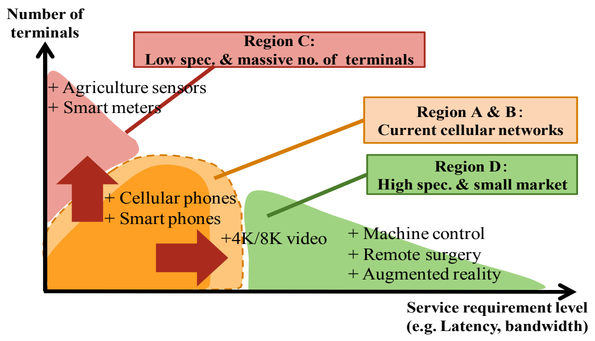

Figure 1 Expansion of the operator's business area

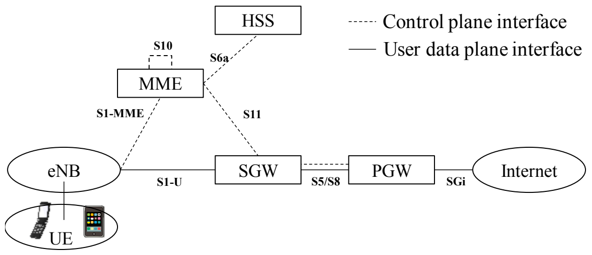

Figure 2 EPC architecture

II. RELATED WORKS

The concept of network slicing was first proposed by the Global Environment for Network Innovations (GENI) initiative [6]. It creates independent end-to-end slices composed of dedicated network topology, virtual nodes, and protocols as a testbed for research. In addition, it allows multiple virtual nodes and networks to be established on the same physical infrastructure. GENI, however, does not consider multiplexing services in one slice or reducing operating cost.

Establishing service-dedicated slices has been studied as a way of providing a simplified EPC with fewer functions or messages. In one study [7], when fixed-position M2M terminals transmit data, eNB transfers the traffic to an M2M-dedicated EPC that authenticates M2M terminals as a group and reduces unnecessary procedures such as paging. Utilizing this dedicated EPC in this manner makes it possible to dramatically reduce authentication signaling costs as the number of M2M terminals increases. To distribute such unique service traffic to the appropriate core network, the DECOR architecture has been standardized in 3rd Generation Partnership Project (3GPP) Rel-13 [2]. DECOR specifies how one or more dedicated core networks within a Public Land Mobile Network (PLMN) can be selected for specific types of subscribers. In another study [8], MME, HSS, SGW, and PGW nodes were virtualized to reduce unnecessary functions, such as location management and hand-over management, for fixed-position M2M terminals. An evaluation of resource consumption] through a simulation showed that this virtualization method reduced CPU usage of each virtualized node. Regarding latency, the current EPC architecture establishes tunnels between eNB and SGW as well as SGW and PGW; it therefore prolongs latency because of the complicated encapsulation/decapsulation procedure for the tunneling and hand-over procedure. Latency can be reduced by centralizing the control-plane functionality of the EPC and utilizing SDN. This configuration is also cost effective as it reduces the number of communication messages [9].

Mobile edge computing (MEC) [10] utilizes a physically close virtual server as an edge server. In the current EPC architecture, all traffic has to pass through the SGW and PGW and then the Internet, and that restriction leads to an increase in latency. MEC enables operators to establish their service on an edge-server in the mobile network flexibly. Hence, it allows them to assure ultra-low latency and high bandwidth as well as real-time access to radio-network information that can be leveraged by applications which requires ultra-low latency.

However, all above-described methods were proposed only in the context of limited service requirements and resource usage, and the service-slice mapping policy they use is static. In other words, the proposed methods do not account for various and dynamic service requirements and traffic patterns to be accommodated by a single network, which is the objective of the 5G-era mobile core.

III. REQUIREMENT FOR SLICE MANAGEMENT ARCHITECTURE

The requirements of a management-architecture for provisioning slicing-based network services were stipulated first. In a similar manner to IaaS technology [11], in the case of network slicing, the network infrastructure is decoupled from the services. Therefore, both the services and network infrastructure need to be provided and managed by different functional entities. A slice is composed of a set of network functions, which run on top of physical machines in the operator's cloud network. Each slice is mapped to one or more services in order to minimize the operational cost; thus, network slicing is an ideal solution for managing different slices and services independently. In fact, the entity for managing network infrastructure also has responsibility to manage physical machines (servers, hypervisors, etc.) and network switches; however, as for the present proposal, it is limited to managing physical infrastructure.

To establish the architecture for managing network slices, it is necessary to investigate the following three points:

- An appropriate abstraction model for slicing-based service provisioning
- Functional entities ranging from designing services to provisioning them on the infrastructure
- Management functions for each entity in the service-provisioning process leveraging NFV/SDN technologies, e.g., ETSI-ISG,NFV end-to-end architectures [10].

IV. SLICING-BASED SERVICE MANAGEMENT ARCHITECTURE

A three-layer slice model containing business entities for providing and operating slices for fulfilling each service requirement and multiplexing two more services in a slice (in order to reduce operational costs) is depicted in Figure 3. In this model, the slice-creation sequence utilizes the DECOR architecture [2].

A. Three-layer slice model

Each slice consists of three layers, a *physical/virtual-resources layer*, a *virtual-network layer,* and a *service-instance layer*. Each layer is managed and configured by an extended Management and Orchestration (MANO) function [4]. MANO is a component designed by the European Telecommunications Standards Institute (ETSI) as part of the NFV specification. It is composed of a virtual-infrastructure manager (VIM), a virtual-network-function manager (VNFM), and an NFV orchestrator (NFVO). The VIM manages the NFV infrastructure (NFVI), which comprises physical- and virtual-hardware resources such as computing, storage, and networking. The virtual network function (VNF) comprises mobile-network function software (e.g., eNB and MME) and switches that can be deployed on the NFVI. The VNFM is responsible for the lifecycle of VNF instances, such as instantiating VNFs, scaling out/in, and auto healing. The NFVO manages multiple VIMs and thereby operates the complete network.

Based on NFV architecture explained above, the three following layers for our slicing concept are defined.

(1) Physical/virtual resource layer

This layer consists of physical and virtual servers and transport switches managed as shared infrastructure resources by a VIM, which includes a software-defined network controller (SDN-C). A complete set of physical and virtual resources is provided for the virtual network layer.

(2) Virtual-network layer

This layer consists of a set of physical and virtual functions such as a service-application function, network nodes, and communication protocols. It has complete physical and virtual resources dedicated for services. The VNFM and NFVO cooperate in allocating and managing nodes for each slice.

(3) Service instance layer

End-user services such as mobile broadband services, smart meters, and remote surgery are managed as independent service instances on the virtual network layer (as shown in Figure 3). The service requirements (e.g., SLA) are monitored and guaranteed by an OSS (operation support system) and a BSS (business support system).

B. Business entities for slice management

Identifying the functional entities involved in managing the life cycle of the slices is a prerequisite for determining their demarcation points, interfaces, and information passed between them. The regions enclosed by dotted lines in Figure 3 represent the following four distinct functional entities of the proposed architecture:

- *Service provider (SP)* defines the service contract with the SO and provides services to end user.
- *Service operator (SO)* determines the "system-performance requirements" and "slice design" based on the contract and sends them to the VNP/O as a request to create a slice. The SO also continuously monitors the QoE (quality of experience) of the services in each slice.
- *Virtual network provider/operator (VNP/O)* creates a slice according to the slice design from the SO. In the slice-creation phase, intelligent service-accommodation logic and function entities are required to multiplex appropriate services for cost-effective operation. The SMF (service-slice mapping function) in the VNP/O determines whether to allocate a service to a slice on the basis of the operator's policy for service accommodation logic. On the basis of the results from the SMF, the VNP/O orders the InP to allocate physical/virtual resources for implementing the appropriate software, and it also continuously monitors the QoS of each slice.
- *Infrastructure provider (InP)* manages the lifecycle of physical and virtual resource and provides a complete set of resources for the slice.

C. Model of service-implementation sequence

The features of network slice management are explained by illustrating the above-described sequences for slice creation and service accommodation. In this illustration, a slice is considered to be composed of dedicated EPC nodes, and a DECOR solution is used [2].

As depicted in Figure 4, when the SP provides new services or changes a service contract, the following nine-step process is executed:

(1) The SP sends a service-instance request to the SO. The SP and SO then make a contract including the function requirements, communication area, data rate, traffic pattern and prediction, communication protocol, and agreement to multiplex the service with other services in a slice.

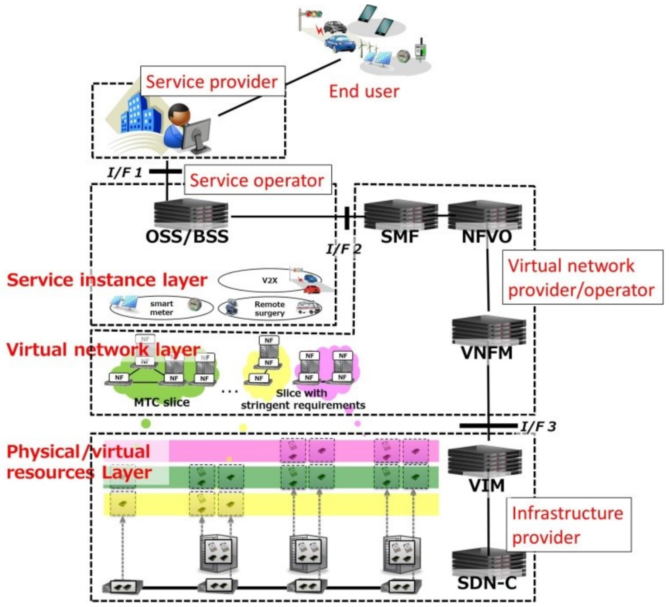

Figure 3: Proposed architecture and business entities

(2) The SO designs system performance requirements and slice design on the basis of the contract.
(3) The SO makes an inquiry to a VNP/O to determine if it can accommodate the service in an existing slice that fulfills the system performance requirements.
(4) The SMF calculates service-slice mapping by applying the algorithm described in Section 4. Then, the results of the calculation are sent to the VNP/O, which designs a slice in detail. In case an already instantiated slice matches the requirements, the process ends; otherwise, the process continues to step (5).
(5) The VNP/O sends an inquiry about reserving physical/virtual resources for the slice to the InP.
(6) The VNP/O implements the functions that fulfill the system performance requirements. In this illustration, the VNP/O creates dedicated MME, SGW and PGW nodes.
(7) The VNP/O configures MME and HSS nodes to redirect the service signaling to new nodes. In this illustration, the VNP/O configures "UE usage type" that the HSS has.
(8) The VNP/O notifies the service operator that it has finished creating the slice.
(9) The UE is firstly attached to the default MME, the MME transmits an authentication request to the HSS, and the HSS returns the configured UE usage type to the MME. On the basis of the UE usage type, the default MME is automatically redirected to the dedicated MME. On completion of these steps, the UE can access the dedicated slice.

V. SERVICE-SLICE MAPPING ALGORITHM IN THE SMF

The proposed service-slice mapping algorithm, which is divided into two phases (i) a *service-slice mapping phase* and (ii) a *resource-expansion phase*, is shown in Figure 5. The first phase is executed with reference to the system performance requirements. The second phase is executed while their traffic pattern, namely, expected number of users and amount of traffic, is taken into account.

(i) Service-slice-mapping phase

This phase consists of the following four steps.

(1) Whether the service requires an isolated slice is confirmed. If it does, a dedicated slice is prepared.

(2) If the service allows multiplexing with other services, the SMF calculates the "function wastage" (w_k) of the slice. Each slice is modeled as $\mathrm{N}_k(s_k, p_k^{NS})$ $(k = 1,2 \dots n)$, where s_k represents the number of services accommodated in the slice, and p_k^{NS} represents the slice's system performance, which is composed of several parameters (e.g., latency, throughput, and UE density). Hence, $p_k^{NS} = [p_{k1}^{NS}, p_{k2}^{NS}, \dots, p_{kl}^{NS}]$ and l represent the number of system-performance-parameters. In the slice design stated in (2) in Section 4, the service's system performance requirements are given as $p^S = [p_1^S, p_2^S, \dots, p_l^S]$. The SMF algorithm calculates w_k at each slice as

$$w_k = \sum_{i=1}^{l} w_{ki} \qquad (1)$$

$$w_{ki} = \begin{cases} p_{ki}^{NS} - p_i^S & (p_{ki}^{NS} \geq p_i^S) \\ s_k(p_i^S - p_{ki}^{NS}) & (p_{ki}^{NS} < p_i^S) \end{cases} \qquad (2)$$

If the slice's system performance is more than the service's performance requirement, the SMF calculates the difference between each parameter as "sub-function wastage w_{ki}". Otherwise, if the system performance of the slice is less than the system performance requirement of the service, w_{ki} is given as the product of the difference between the two parameters and the number of services in the slice. When system performance in a slice increases to fit the service's system performance requirements, all the services in the current slice will be affected. The function wastage of slice k is the sum of all w_{ki}. Note that in this study, the nature of system performance and service performance requirement was not clarified and all function levels were not normalized as dimensions such as CPU usage, energy consumption, and amount of messages.

(3) w_k values of all the slices are sorted, and w_{min} is defined as the minimum w_k. Then, w_{min} is compared with a predefined threshold *th*. If w_{min} exceeds *th*, the SMF creates a new slice for the service; otherwise, SMF accommodates the service in slice *k*. For example, the EPC has *th* of infinity (i.e., all services are multiplexed into one slice and function wastage increases), whereas a slice-per-service architecture, which prepares specific slices for every service has *th* of zero.

(4) If the system performance of the slice is increased, the function wastage of some services will exceed the threshold. In such a case, appropriate slices for those services are reallocated. Steps (2)-(4) are repeated until the function wastage of each service accommodated in any slice is less than the threshold.

(ii) Resource-expansion phase

In case the SMF does not create a new slice for the service, after an appropriate slice for the service is selected, the SMF calculates the required amount of additional resources on the basis of service traffic information, number of users, traffic pattern of the service, and statistical multiplexing gain of the slice. Then, the SMF orders MANO to scale-up the slice's resources in proportion to the calculation result.

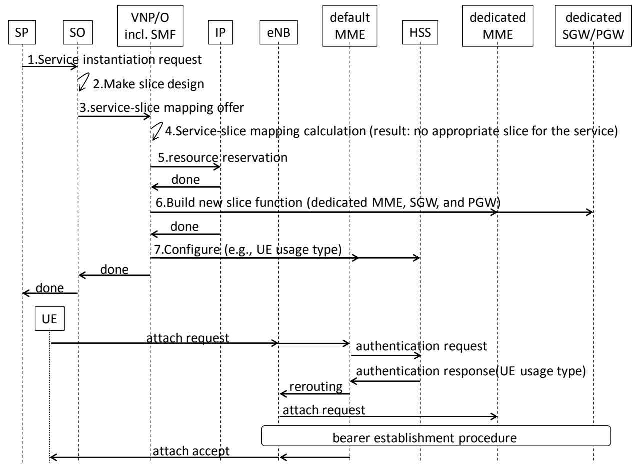
SP
SO
VNP/O incl. SMF
IP
eNB
default MME
HSS
dedicated MME
dedicated SGW/PGW
1.Service instantiation request
2.Make slice design
3.service-slice mapping offer
4.Service-slice mapping calculation (result: no appropriate slice for the service)
5.resource reservation
done
6.Build new slice function (dedicated MME, SGW, and PGW)
done
7.Configure (e.g., UE usage type)
done
done
UE
attach request
authentication request
authentication response(UE usage type)
rerouting
attach request
bearer establishment procedure
attach accept

Figure 4: Service-mapping sequence based on décor.

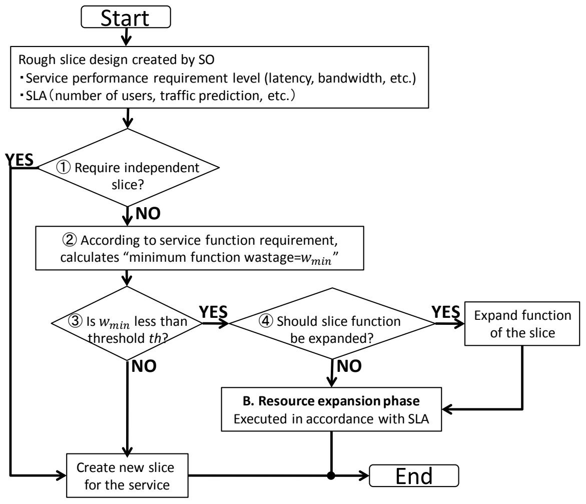
Start
Rough slice design created by SO
・Service performance requirement level (latency, bandwidth, etc.)
・SLA (number of users, traffic prediction, etc.)
① Require independent slice?
YES
NO
② According to service function requirement, calculates "minimum function wastage=w_{min}"
③ Is w_{min} less than threshold th?
YES
NO
④ Should slice function be expanded?
YES
NO
Expand function of the slice
B. Resource expansion phase
Executed in accordance with SLA
Create new slice for the service
End

Figure 5: Service-slice mapping algorithm

VI. EVALUATION

A. Proposed algorithm

As terms of the system performance levels and system performance requirement levels listed in Tables 1 and 2, respectively, the proposed service-slice-mapping algorithm was evaluated, and the relationship between the threshold values and function wastage for the 24 use cases and four system performance requirement parameters mentioned in the NGMN whitepaper was determined [3]. Three threshold values, two, three, and four, were used. The order of service initiation requests might change the system performance of the corresponding slice. Therefore, the arrival order was randomly changed, and 10 iterations of service-slice mapping were performed. For comparison of proposed service-slice mapping method and conventional method EPC (in the case of an infinite threshold, i.e., any type of services can be accommodated in an EPC slice) and slice-per-service architecture (in the case of a threshold of zero, i.e., only one type of service can be accommodated in one slice) were evaluated.

Number of slices and function wastage in proportion to *th* are plotted in Figure 6. As shown in the figure, for threshold values of two or more, the proposed algorithm reduces the number of slices, thereby reducing function wastage. On the other hand, the amount of function wastage increases as the threshold increases. These results indicate that there is a tradeoff between number of slices and function wastage. Accordingly, the proposed algorithm provides the network operator with a means to select an appropriate threshold that balances the number of slices (namely, satisfies the system-performance requirements of the services) and resource wastage.

B. Effect of varying number of slices

To quantify resource usage in comparison with slice-per-service architecture, the difference between CPU usages of a host server based on the number of VMs was evaluated on our testbed. In particular, 20 services with the same traffic patterns were defined. It was emulated as continuous sensor data traffic (number of users: 100; packet-transmission rate: 2 packet/s; packet size: 128 bytes; data rate: 2 kbps; packet direction: from the client to the server only). The number of slices that can accommodate traffic of 20 services was one (i.e., a slice number of one means all services are accommodated in the same slice, two means 10 services per slice, four means five services per slice, ten means 2 services per slice, and 20 means one service per slice). Each slice is composed of current EPC nodes utilizing OpenEPC [12]. The MME/HSS nodes and SGW/PGW nodes are hosted in different VMs running on the same physical server.

CPU usages of SGW and PGW nodes as well as the behavior of the physical server were measured. The measurements were performed 120 times per second, and averaged over three trials. When the number of slices processing the service traffic was more than one, the mean CPU usages of each server in the slice were multiplied.

Relative CPU usages of the SGW and PGW nodes and their host physical server for multiple slices are shown in Figure 7. Although the total amount of user traffic processed is the same for each number of slices, for 20 slices, the SGWs and PGWs consume approximately double the amount of CPU resources compared to that of in the case of one slice.

Table 1: System-performance levels

parameter / degree	0	1	2	3
latency(ms)	100~	~100	~10	~1
mobility(km/h)	N/R	~30	~200	200~
throughput(Mbps)	~1	~10	~50	50~
UE density(km^2)	N/R	~400	~2500	2500~

Table 2: System-performance-requirement levels

use case family	service	latency (ms)		mobility (km/h)		throughput (Mbps)		UE density (km^2)	
①Broadband access in dense areas	1. Pervasive video	10	**2**	100	**2**	300	**3**	2500	**2**
	2. Operator cloud service	10	**2**	100	**2**	300	**3**	2500	**2**
	3. Dense urban society	10	**2**	100	**2**	300	**3**	2500	**2**
	4. Smart office	10	**2**	5	**1**	1000	**3**	75000	**3**
	5. HD Video sharing in stadium	10	**2**	5	**1**	50	**2**	150000	**3**
②Broadband access everywhere	6. 50 Mbps everywhere	60	**1**	30	**1**	50	**2**	400	**1**
	7.Ultra-low cost network	100	**0**	50	**2**	10	**1**	16	**1**
③High user mobility	8. High speed train	10	**2**	500	**3**	50	**2**	2000	**2**
	9. Moving hotspots	10	**2**	500	**3**	50	**2**	2000	**2**
	10. Remote computing	10	**2**	200	**2**	50	**2**	2000	**2**
	11.3D connectivity	10	**2**	1000	**3**	15	**1**	0.26	**1**
④Massive IoT	12. Smart wearables	100	**0**	30	**1**	0.1	**0**	200000	**3**
	13. Sensor networks	100	**0**	0	**0**	0.1	**0**	200000	**3**
	14. Mobile video surveillance	10	**2**	120	**2**	300	**3**	2500	**2**
⑤Extreme real-time communication	15. Tactile internet	1	**3**	5	**1**	50	**2**	N/R	**0**
⑥Lifetime Communication	16. Natural disaster	100	**0**	500	**3**	1	**1**	10000	**3**
⑦Ultra-reliable communication	17. Automated traffic control	1	**3**	500	**3**	10	**1**	N/R	**0**
	18. Collaborative robots	1	**3**	100	**2**	10	**1**	N/R	**0**
	19. Remote object manipulation	1	**3**	500	**3**	10	**1**	N/R	**0**
	20. e-health	10	**2**	500	**3**	10	**1**	N/R	**0**
	21. Public safety	100	**0**	500	**3**	10	**1**	N/R	**0**
	22.3D connectivity/ drones	10	**3**	500	**3**	10	**1**	N/R	**0**
⑧Broadcast like services	23. News and information	100	**0**	25	**1**	50	**2**	N/R	**0**
	24. Broadcast like service	100	**0**	25	**1**	50	**2**	N/R	**0**

Extra processing is required when the number of slices increases, but the resource wastage is increased due to the larger number of processing elements. For example, one slice requires one SGW node and one PGW node (i.e., 1 SGW and 1 PGW), since all services are served by a single slice. On the contrary, 20 slices require 20 SGWs and 20 PGWs, because 20 different slices are mapped to 20 different services. In addition, each SGW and PGW requires some application-level processing, which increases the consumption of CPU resources.

This trend (i.e., increasing resource wastage) is more conspicuous in Figure 8, which shows CPU usage (excluding SGW/PGW operation) of the physical server. The CPU usage is generated by the management tasks in the hypervisor and host OS, which need to manage more VMs when the number of slices increases. This result validates that resources are wasted because of the complexity involved in managing numerous slices.

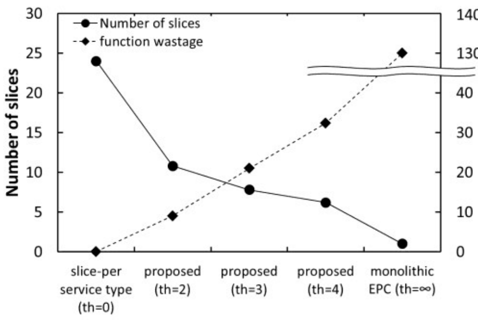

Figure 6: Number of slices and function wastage

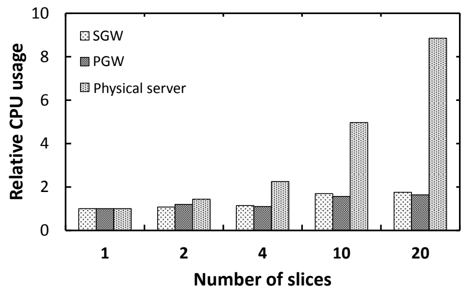

Figure 7: Relative CPU usage

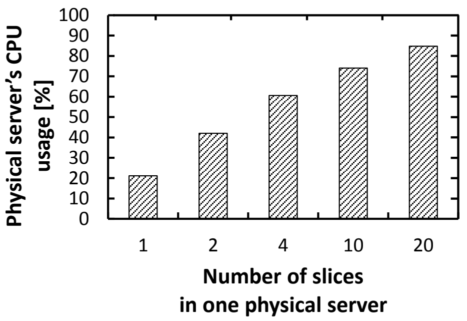

Figure 8: Physical server's CPU usage excluding SGW/PGW operation

VII. CONCLUSION

The proposed architecture and service-slice mapping algorithm determines whether to create a slice or multiplex a service into one of the operating/existing slices on the basis of the system performance requirements and current slice's status. The results of an experimental evaluation of the algorithm show that the proposed algorithm reduces both the number of slices and function wastage while selecting appropriate parameters. In addition, to highlight the efficiency of our algorithm, the effect of loss of multiplexing gain at SGWs, PGWs, and the physical server was evaluated on the testbed. The results of the emulation show that under the same traffic scenario, more CPU resources are needed as the number of slices increases. Especially, CPU resources of the physical server are increased drastically when the management tasks in the hypervisor and the host OS are executed. In other words, reducing the number of slices by utilizing the proposed algorithm effectively reduces operation cost. Our future work includes quantifying the thresholds for function wastage and slice creation. In this paper, four performance requirement levels were defined; however, in practice, these parameters will vary depending on the actual functions for each service. Normalizing these parameters into several dimensions (such as resource usage) will enable the function wastage to be calculated more precisely. There is a tradeoff between number of slices and total amount of function wastage; therefore, the optimum balance between these parameters should be identified.

REFERENCES

[1] NTT DOCOMO, "5G white paper," [Online] Available at: https://www.nttdocomo.co.jp/english/binary/pdf/corporate/technology/whitepaper_5g/DOCOMO_5G_White_Paper/pdf

[2] 3GPP TS 23.401 V13.6.1: "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access," 2016.

[3] NGMN, "5G white paper," [Online] Available at: https://www.ngmn.org/5g-white-paper/5g-white-paper.html

[4] ETSI GS NFV 002 V 1.2.1 "Network Functions Virtualisation (NFV); Architectural Framework," 2014

[5] N. Mckeown, et.al, "Openflow: Enabling innovation in campus networks," ACM SIGCOMM, vol. 38, pp. 69-74, Apr. 2008.

[6] M. Berman, et.al, "GENI: A federated testbed for innovative network experiments," IEEE Journal of Computer Networks, vol. 61, pp. 5-23, Mar 2014.

[7] T. Taleb, et.al, "Lightweight mobile core networks for machine type communications," IEEE Access, vol. 2, pp. 1128-1137, Sep 2014.

[8] H. Baba, et.al, "Lightweight virtualized evolved packet core architecture for future mobile communication," IEEE WCNC, pp. 1811-1816, Mar 2015.

[9] S. Chourasia, et.al, "SDN based evolved packet core for efficient user mobility support," IEEE Network Softwarization, pp. 13-17, 2015

[10] ETSI, "Mobile edge computing whitepaper," [Online] Available at: http://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf

[11] L. M. Vaquero, et.al, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Computer Communication Review, vol. 39, pp. 50-55, January, 2009

[12] The OpenEPC Project, "http://www.openepc.com"