Mobile Service Continuity for Edge Train Networks

Osamah Ibrahiem Abdullaziz*, Samer T. Talat†, Chen-Hao Chiu‡ and Li-Chun Wang§
*§Department of Electrical and Computer Engineering, National Chiao Tung University, Taiwan
*yabolahan.04g@g2.nctu.edu.tw, §lichun@g2.nctu.edu.tw
†‡Information and Communications Research Laboratories, Industrial Technology Research Institute, Taiwan
†talat@itri.org.tw, ‡chenhaochiu@itri.org.tw

***Abstract*—In moving train networks, two-hop architecture is adopted to improve users experience by reducing the interaction between on-board users and base stations on the train route. In addition, edge networking have emerged as a solution for bringing services to the proximity of the users. However, deploying two-hop and edge networks do not guarantee a continuous service delivery for train users. When a large number of users transit from the train to the land, they experience service interruption due to control signalling storm and backhaul latency. In this paper, we propose a holistic edge service management system to provide mobile service continuity. The contribution of this paper is twofold. First, we develop an enhanced handover scheme that reduces control signals by handling user mobility at the edge. Second, we develop a pre-copy migration scheme that eliminates backhaul latency by relocating containerized applications to the user proximity across edge train networks. Our experimental results show that the two proposed solution can reduce the control signals and migration downtime by 50% and 36%, respectively.**

***Index Terms*—Group handover, Container migration, Edge networking, Service continuity.**

I. Introduction

In recent years, high-speed rails have been deployed across many countries around the world. For instance, 2000 km and 10,000 km of train lines are to be deployed by 2020 in France and Spain, respectively [1]. In such moving infrastructures, several challenges are faced due to mobility such as complex signal processing, signal penetration loss, frequent handover (HO) and terrestrial signal blocking. To overcome these challenges, two-hop architecture solution is adopted to provide better signal quality for on-board users [2] [3]. In two-hop architecture, on-board small cells are deployed behind customer-premises equipment (CPE) and connected to the roadside base stations. Consequently, there is no direct interaction between on-board users and roadside base stations, which significantly reduces the number of handover events.

In train networks, users can enjoy a variety of mobile services, such as video conferencing and online gaming. To enhance user experience, multi-access edge computing (MEC) and network function virtualization (NFV) have emerged as solutions where the cloud services are brought closer to the edge of the network. On one hand, MEC virtualizes applications at the edge of the network and reduces the end-to-end delay. On the other hand, NFV decouples the network functions and applications from the underlying hardware and allows them to be implemented as software. Now that mobile services can be deployed independently from the hardware

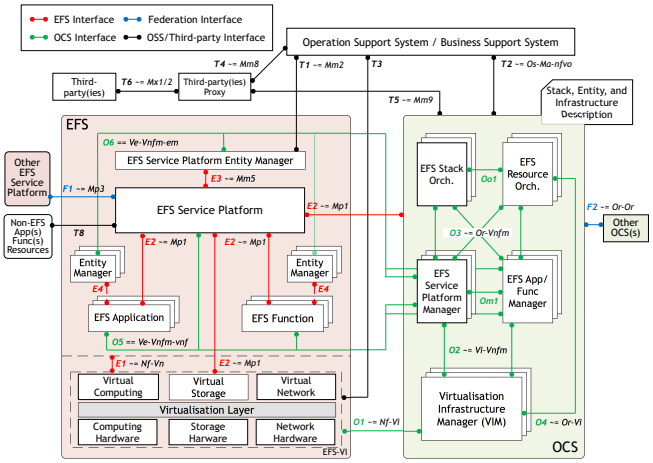


Figure 1: 5G-CORAL reference architecture [7]

in a distributed fashion, software-define networking (SDN) enables a dynamic and responsive network to new services. SDN complements MEC and NFV especially due to the separation of the control and data planes which simplifies management, provides programmability and enhances scalability and performance.

Together, MEC, NFV and SDN technologies empower operators to efficiently orchestrate and scale their networks. A novel integration of these technologies is demonstrated by 5G-CORAL [4] which leverages edge and fog computing to create unique opportunity for access convergence. It contemplates ETSI NFV [5] and ETSI MEC [6] standards while also considering mobile and volatile computing, networking and storage resources in a multi-RAT environment. 5G-CORAL architecture consists of two major building blocks as shown in Fig. 1. The edge and fog computing system (EFS) contains edge and fog resources that belong to a single administrative domain. An EFS provides service platforms, functions, and applications on top of virtual resources, and may interact with other EFS domains. The orchestration and control system (OCS) is responsible for managing and controlling the EFS. In this work, we adopt 5G-CORAL architecture to improve user experience in edge train networks.

Although the two-hop architecture and edge networks can enhance user experience, we recognize two issues in regards to mobile service continuity namely, control signalling storm [8] [9] and backhaul latency. Firstly, when a large number of train users transit from on-board a train to a station, they generate a signalling storm due to massive HO

events. This massive signalling can lead to connection teardown thus affecting users experience. Secondly, also due to mobility, users leave the initial serving edge (i.e., on-board the train) which leads to increased latency due to backhauling.

The contributions of this work are summarized as follows:

- Propose a group HO scheme which exploits NFV and MEC by deploying virtualized mobility management entity (vMME) at the edge to lower signalling storm.
- Develop container pre-copy migration scheme to relocate applications in MEC environment to eliminate backhaul latency. This work outperforms the state-of-art (SoA) container migration scheme [10].

The rest of the paper is organized as follows. Section II presents the related work in the area of inter-MME HO and application migration. Sections III and IV present the mobile service continuity solution by introducing the proposed inter-MME HO and edge application migration, respectively. The experimental results are discussed in Section V. Finally, we draw our conclusions in Section VI.

II. Related Work

A. Inter-MME Handover

In LTE, mobility management entity (MME) is the main function that handles mobility control signalling. It is responsible for initiating paging and authentication of the user devices. Also, MME retains location information at the tracking area level for each user and then selects the appropriate gateway during the initial registration process. Inter-MME Ho is needed especially if MME is adopted in moving networks. Inter-MME HO involves three control stages [11]. First, the source eNB initiates the HO by sending a request message over the S1-MME reference point. Second, the source MME selects the target MME and configures a messaging tunnel over a control interface called S10. Finally, the source MME transfers the configuration message to target eNB over S1 interface.

Several works have been proposed to improve the HO performance [12]–[14]. For example, a multicast paging procedure to alleviate the MME signalling load is proposed in [12]. Also, the MME performance with multicast and unicast paging procedures is measured. In [13], SDN is utilized to evaluate the HO performance and to reduce the jitters and packet loss. In [14], SDN is used to improve the HO performance. In [15], sharedMEC is proposed to support user mobility and reduce total cost of migration among MEC during HO. However, none of the existing works addressed the signalling reduction of S10 interface during inter-MME HO. For that, we propose an enhanced Inter-MME HO scheme which initiates the HO process for QoS-based classified group before it is required.

B. Containerized Application Migration

A container is an isolated process running on top of a shared kernel. There are two types of containerization technologies, namely system-based and application-based containerization. On one hand, system containers behave like a standalone Linux system. That is, the system container, such as Linux Container (LXC) has its own root access, file system, memory, processes, networking and also can be rebooted independently from the host system. On the other hand, the application container isolates an application from other applications running on the same host kernel and operating system. This means that the development of a containerized application with necessary libraries, configurations and dependencies does not affect other applications and also the host system.

Container migration can be classified into stateful and stateless. In stateless migration (also known as cold or offline migration), the state of the container is not preserved when the container is moved to the destination host. In the case of stateful migration (also known as live migration), the state of the container is retained when the container is resumed at the destination host. There are three types of stateful migration schemes in the literature as follows:

- *stop-and-copy* - freezes the container, checkpoints its state, copies the container image and its state to the destination then restores the state form the checkpoint [16].
- *pre-copy* - performs iterative state checkpointing while the container is running till the amount of in-memory change is at minimum, then concludes with a shorter stop-and-copy [17]. Iterative checkpointing reduces the size of the final checkpoint which is performed while the container is frozen. This minimizes the time required for the final checkpoint and the time required to copy the checkpoint to destination.
- *post-copy* - performs a short stop-and-copy to move essential state data, then starts the container at the destination and retrieves the rest of the data when required [18]. This type of migration has a very small downtime but containers may suffer from performance degradation due to the time needed to wait for the requested memory pages.

In traditional hypervisor-based virtualization, virtual machine (VM) migration is well investigated [19] and many successful solutions are commercially available. For instance, a pre-copy based VM migration scheme is presented in [17]. An active VM continues to run in the course of in-memory data iterative pre-copying. During a consecutive iteration, only changed memory (dirty pages) are transferred. At last, a final state copy is performed while the VM instance is frozen and then transferred to the destination host. This way, the amount of downtime is greatly reduced when compared to a pure stop-and-copy scheme. Although VM migration is a mature technology, it relies on hypervisors and most of the existing solutions are tailored for data centre environment where network-attached storage (NAS) and specific virtualization technology are utilized. NAS enables all the host machines in a data centre to access a network-shared storage which reduces the time spent during the copying stage. However, in a scenario where migration takes place between MECs (train to land), state and local-disk storage must also migrate over wide area network.

Container migration has lately caught much attention from the research community [20] [10]. Especially, since containerization offers many advantages, in terms of resource efficiency and performance, over traditional hypervisor-based virtualization. This fact enables the instantiation of lightweight containerized applications suitable for IoT ser-

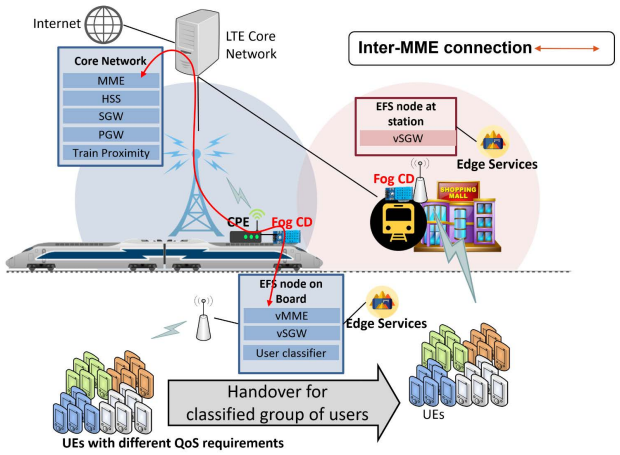


Figure 2: Enhanced inter-MME handover in two hop architecture

vices [21]. In [20], container migration mechanism is developed for power efficiency optimization in heterogeneous data center. This work assumes that the source and destination hosts have access to a NAS and thus container data is not copied over wide area network. Recently, a framework for migrating containerized applications is presented in [10]. The proposed framework is the first to consider container migration for MEC environment. Fundamentally, the framework is a layered model that aims to reduce the migration downtime. While the presented results show reduction in downtime, the framework relies on stop-and-copy which is inefficient method for containers with large in-memory state. In our proposed solution, we develop a pre-copy migration scheme to relocate both system-based and application-based containers across edge networks.

III. Enhanced Inter-MME Handover

In this paper, we deploy the vMME on-board as illustrated in Fig. 2. As mentioned, the S10 interface will face large amount of control signals when hundred of users devices perform HO simultaneously. Therefore, we enhance the S10 interface to reduce the signalling between on-board and on-land EFS nodes, where the adopted EFS virtualization infrastructure has the MME functionality. It is the totality of the hardware and software components that build up the environment. In particular, the EFS consists of several elements namely applications, functions and services as shown in Fig. 3. These elements are detailed as follows:

- *User Applications*: Containerized applications are created for UEs based on their QoS. For example, a containerized video streaming application is created for a group of UEs demanding the same QoS requirements.
- *User Classifier function*: The UEs on-board have different QoS requirements. The user classifier function classifies the UEs into groups based on context information such as QoS Class indicator (QCI) and allocation and retention priority (ARP) [22] that are extracted from vMME (i.e. nine different QoS types defined in the legacy system). The classification is accomplished by two steps. In the first step, the user classifier extracts UEs context information and sorts them based on QCI and ARP index. Tn the second step, the classifier shares the groups with vMME in descending QoS order for service HO triggering (see Algorithm 1). The classification process is very important to reduce the signalling overhead during the inter-MME Handover and maintain the service interruption at the minimum.
- *vMME*: vMME interacts with target MME, located in the core network, to perform the enhanced Inter-MME HO as shown in Fig. 4. In step 0 to step 10, the enhanced inter-MME HO is executed ahead of time unlike the legacy system (i.e. inter MME handover) as follow: As the train approaches the station, MME detects it and sends a trigger to vMME. Next, vMME executes group HO based on the user classification. In particular, the vMME transmit essential UEs context information early which will reduce the amount of signalling during the HO process.Then, vMME forwards the relocation UEs information request to target MME and receives its response. The remaining part follows the standard inter-MME HO procedure [11].
- *Train Proximity*: The proximity of trains is handled by the MME of the core network. The MME monitors several UEs and PCIDs (eNB ID), and since the route path of the train is known, the approximate location of the train can be estimated. In particular, the MME monitors the CPE on-board connection to eNB in the two-hop architecture. When the train is approaching the station, the CPE will HO to a specific eNB near the station. Thus, MME can estimate the approximate location of train and executes the HO triggering function. In HO triggering function, MME sends the trigger to the vMME ahead of time. Also, the user containerized application will migrate from on-board and on-land EFS nodes.

Then, it executes service HO triggering as UEs are moved to specific eNB.

Algorithm 1: Service HO Triggering

Input: Active UE list UEs ($UEs = UE1, UE2, ..., UE_i$), each UE_i has QCI_i and ARP_i

Output: the selected UEs H to be handovered

Step 1: Sort UEs into group list G_j, $1 \leq$ j ≤ 9

for *each* UE_i *in* UEs **do**
  G_{QCI_i}.append(UE_i)

for *each group list* G_j, $1 \leq j \leq 9$ **do**
  SortByARP(G_j)

Step 2: Decide H and triggers

Concatenate G_i, $1 \leq$ j ≤ 9 to full list G

while $G.len() > \Theta$ **do**
  H = G.deque(Θ)
  SendServiceHOTrigger(H)

H = G.deque(G.len())

SendServiceHOTrigger(H)

IV. Edge Application Migration

The proposed migration scheme enabling technologies include 5G-CORAL, Linux containers, checkpoint and restore in user space (CRIU), and remote file synchronization (*rsync*). The orchestrator part of the OCS manages

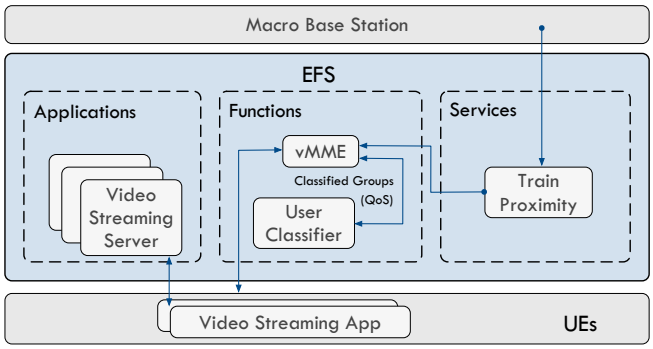


Figure 3: EFS entities for train network

the lifecycle of the containerized applications running on the EFS nodes. It supports management operations such as instantiating, cloning, migrating, scaling and terminating. CRIU is utilized to checkpoint the state of the migrating containers. The local-disk and the state of the containers are copied to the destination node by using *rsync* for its remarkable speed and efficiency.

To relocate a container between EFS nodes with minimal downtime, we develop a pre-copy migration scheme. Fig. 4 shows the pre-copy procedure as an integral part of the proposed mobile service continuity solution. The logical steps of the proposed scheme are summarized in following:

Step 1: *Local-disk-copy* - the container base-image is assumed to be available in all edge nodes to reduce traffic overhead and to keep the total migration time to minimal. Local-disk synchronization is performed to copy application related files.

Step 2: *Iterative-pre-copy* - the container state is dumped to the source node storage, and then copied over to the destination while the container continues to run. Next, pre-copy iterations are performed to checkpoint and copy only the memory pages that have changed (dirtied) since the last checkpoint.

Step 3: *stop-and-copy* - the container gets frozen in this step, and then a final checkpoint and copy round is performed. The downtime observed by the user occurs during this step.

Step 4: *Restore-and-terminate* - the container is restored in the destination EFS node and the frozen container in the source node gets terminated.

It is worth noting that checkpoint and restore functions of CRIU are computationally expensive. The checkpoint function collects a process tree and its resources, freeze the process, then write them to files. The restore function reads the files, resolves shared resources, fork the process tree then restore the process resources. Both functions perform I/O operations which are generally slow especially on rotational block devices such as hard disk drive (HDD). To improve the migration scheme, we include the following enhancements on the EFS nodes:

- *Low-latency computing capabilities*: Linux generic kernels fail to provide time guarantees for time-critical applications. Hence, we incorporate low-latency computing into the EFS nodes. In addition, we scale the CPU performance to avoid latency caused by waking up from idle state.

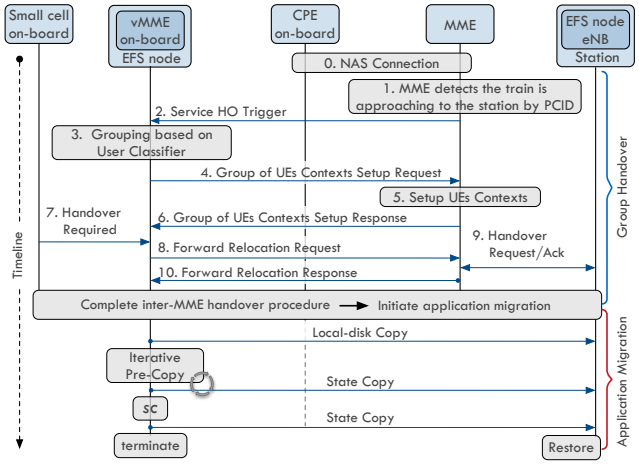


Figure 4: Service continuity flowchart

- *Fast storage*: HDD uses mechanical mechanism to persistently store data in blocks of 512 bytes. As such, I/O operations experience seeking time delays (i.e., the time it takes the disk head to find the target track). Here, we utilize temporary file system (*tmpfs*) to enhance the performance as it allows short-term files to be written and read without generating disk I/O.

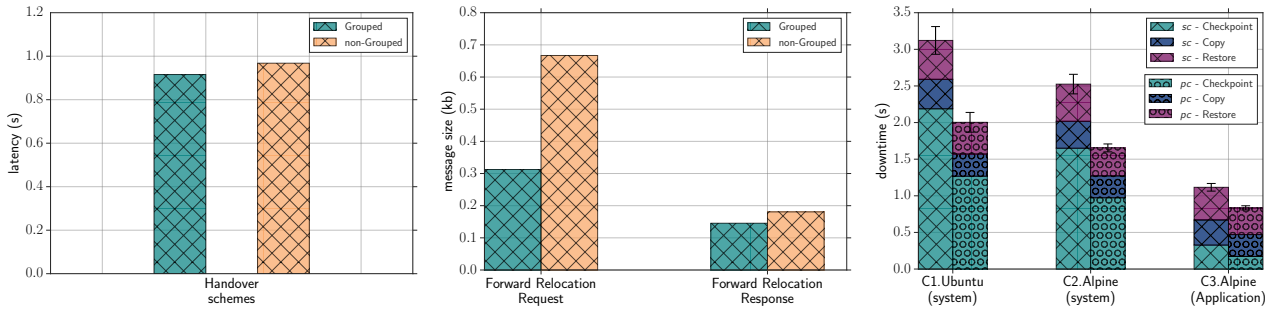
V. Experimental Results

In this section, we explain the experimental setup for the two-hop architecture and the EFS deployment on-broad and OCS deployment on-land. Then, we present our experimental results for the enhanced inter-MME HO and the edge application migration. In particular, the control signal packet sizes, HO latency, downtime measurements for legacy system and the proposed schemes are highlighted in the following subsections.

A. Experimental Setup

In the emulation environment of moving network, we utilize real-time reference signals received power (RSRP) and received signal strength indication (RSSI) from a CPE of Taiwan high-speed rail. The collected signals are taken from a 30 km route at an approximate train speed of 300 km/h. The emulation environment consists of the following components:

- *EFS Node*: Hosts several applications, functions and services such as video streaming application, vMME and user classifier function, and train proximity service, respectively. In our experiment, we utilize NextEPC [23] framework as baseline for vMME. Then, we modify vMME to fit our proposed scheme.
- *Small cell*: A small cell acts as an eNB on-board of the train. Another small cell acts as an on-land Macro base station that has the capability to adjust RSSI and RSRP to emulate the train signal degradation.
- *CPE*: Gateway between on-board and on-land small cells. It can also act as WiFi access point.
- *UE*: On-board user devices are emulated using NextEPC UE emulator. The emulated users have different services requirements.



(a) (b) (c)

Figure 5: Group handover and application migration. (a) Handover latency. (b) Handover control messages overhead. (c) Container migration downtime.

Table I: Experimental Setup latency and throughput

Measurement	Avg. Value
latency between EFS node and core network	50ms
Ping latency between UE to core network	72ms
iperf tcp test (EFS node to core network)	35Mbps
iperf tcp test (UE to core network)	15.3Mbps

- *Core Network and OCS*: A server hosts the core network and emulates the train proximity. Also, runs the OCS to enable application migration based on the train mobility.

The experiment is carried out as follows. First, the MME, the small cell and the CPE are powered on. Then, the vMME is powered on and connected to HSS in the core network. Next, four emulated UEs connect to the small cell (on-board). At the core network, the MME connects to the target eNB (on-land). As train mobility is emulated, the MME sends HO trigger to vMME which in turn classifies the users into groups based on the QoS. In this emulated train network, latency and throughput measurements are executed as shown in Table I.

B. Group Handover Results

Fig. 5 (a) represents a comparison between the enhanced and the legacy inter-MME HO. The *x*-axis represents the enhanced inter-MME (Grouped) and legacy inter-MME (non-Grouped) while the y-axis represents the inter-MME HO time. The grouped HO slightly improved the total HO time when compared to non-grouped inter-MME HO. In both cases, the HO time is high due to the emulation environment overhead. In real deployment, the average HO time is approximately 200ms. It is important to noted that the latency reduction of HO is not the focus of this work. As such, maintaining the same HO latency highlights our objective to reduce signalling storm.

Fig. 5 (b) represents a comparison between grouped and non-grouped inter-MME HO control messages. The *x*-axis represents the forward relocation request and forward relocation response, respectively. On one hand, the forward relocation request contains several context information request such as International mobile subscriber identity (IMSI), target identification, and UE Time. On the other hand, the forward relocation response provides the reply for the requested context information for UEs. The *y*-axis represents the average control message sizes in bytes. In case of forward relocation request, the grouped inter-MME scheme scaled down the average control messages up to 50% per user in comparison to the non-grouped scheme. That is, the forward relocation request messages size per UE are 690 and 320 bytes for non-grouped and grouped schemes, respectively. As such, a significant signal reduction is achieved when hundred of UEs are managed by the proposed scheme. For the forward relocation response, the grouped inter-MME HO scheme reduces the overhead by 25% per user in comparison to the non-grouped scheme. The forward relocation response messages are 207 bytes and 149 bytes per user for non-grouped and grouped, respectively. Note that this reduces the signals to core network significantly in large scale scenario and at the same time contributes towards application stability at the end user side.

Table II: Downtime of migration tasks.

Stop-and-copy (*sc*)			Pre-copy (*pc*)		
Checkpoint	Copy	Restore	Checkpoint	Copy	Restore
C1 - System Container (Ubuntu 4.4.0-116)					
2.18s	0.40s	0.53s	1.26s	0.30s	0.43s
C2 - System Container (Alpine 3.7)					
1.65s	0.36s	0.50s	0.97s	0.29s	0.38s
C3 - Application Container (OCI - Alpine 3.7)					
0.32s	0.34s	0.44s	0.17s	0.29s	0.36s

C. Application Migration Results

To benchmark container migration, we implemented the stop-and-copy (*sc*) scheme to reproduce the results presented in [10] and evaluated its downtime against our proposed pre-copy (*pc*) migration scheme. The migration experiments were carried out between two EFS nodes. In this experiment, we evaluate the downtime during the migration of a video streaming application running in LXC system containers (Ubuntu and Alpine) and LXC application container (Alpine). The presented results are based on average values of ten trails for each presented case. Fig. 5(c) compares the migration downtime of the two schemes. Since the rate of dirty pages for the video streaming application is minimal, most of the downtime is attributed to the common steps (i.e., final checkpoint $\rightarrow$ state copy $\rightarrow$ restore) of both migration schemes rather than the amount of data copied. For instance, in the case of C1 (Ubuntu system container), the downtime due to the checkpoint process are $2.19s$ and $1.27s$ for the *sc* and *pc*, respectively. In the case of C3 (Alpine application container), the observed migration downtime is an average of $0.83s$. Table II shows the breakdown of the downtime during

migration tasks. Overall, the proposed *pc* migration scheme with the EFS enhancements reduces the downtime by 36% when compared to the SoA container migration.

VI. Conclusions

In moving infrastructure scenarios such as train networks, two-hop architecture is adopted to improve on-board user experience by reducing the interaction with on-land base stations. Furthermore, edge networks and virtualization technologies can be utilized to bring services closer to the traveling users. Nevertheless, when large number of users transit from train to station, a signalling storm and backhaul latency become challenges to maintain a continuous service. In this paper, we propose a mobile service continuity solution which includes a group handover and application migration schemes. Our experimental results show that the proposed schemes can reduce the control signals and migration downtime by 50% and 36%, respectively.

Acknowledgment

This work has been partially funded by the H2020 collaborative Europe/Taiwan research project 5G-CORAL (grant no. 761586). This research is also partially supported by the Ministry of Science and Technology, under the Grant Number MOST 108-2634-F-009-006 - through Pervasive Artificial Intelligence Research (PAIR) Labs, Taiwan.

References

[1] R. Vickerman, "High-speed rail in europe: experience and issues for future development," *The annals of regional science*, vol. 31, no. 1, pp. 21–38, 1997.

[2] H.-W. Chang, M.-C. Tseng, S.-Y. Chen, M.-H. Cheng, and S.-K. Wen, "Field trial results for integrated WiMAX and radio-over-fiber systems on high speed rail," in *22nd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2011, pp. 2111–2115.

[3] C. Lai, S. B. Chundrigar, S. T. Talat, and H. Chang, "Staying connected on the go : Network sharing based on multioperator traffic inside moving transportation," *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 97–104, March 2018.

[4] 5G-CORAL H2020 project. [Online]. Available: http://5g-coral.eu/

[5] N. F. V. ETSI, "Network function virtualisation architectural framework," *ETSI GS NFV*, vol. 2, p. v1, 2013.

[6] M. ETSI, "Mobile-edge computing," *Introductory Technical White Paper*, 2014.

[7] 5G-CORAL, "D3.1: Initial design of 5G-CORAL orchestration and control system," *Online: http://5g-coral.eu/wp-content/uploads/2018/06/D3.19802.pdf*, 2018.

[8] E. Grigoreva, J. Xu, and W. Kellerer, "Reducing mobility management signaling for automotive users in lte advanced," in *2017 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, June 2017, pp. 1–6.

[9] N. S. Networks, "Signaling is growing 50% faster than data traffic," in *White Paper*, 2016.

[10] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Communications*, vol. 25, no. 1, pp. 140–147, 2018.

[11] G. T. 23.401, "3gpp ts 23.401, general packet radio service (gprs) enhancements for evolved universal terrestrial radio access network (e-utran) access."

[12] I. Widjaja, P. Bosch, and H. L. Roche, "Comparison of mme signaling loads for long-term-evolution architectures," in *2009 IEEE 70th Vehicular Technology Conference Fall*, Sep. 2009, pp. 1–5.

[13] H. T. Larasati, F. H. Ilma, B. Nuhamara, A. Mustafa, R. Hakimi, and E. Mulyana, "Performance evaluation of handover association mechanisms in sdn-based wireless network," in *2017 3rd International Conference on Wireless and Telematics (ICWT)*, July 2017, pp. 103–108.

[14] F. Laassiri, M. Moughit, and N. Idboufker, "An improvement of performance handover in worldwide interoperability for microwave access using software defined network," in *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 12, January 2018.

[15] W. Nasrin and J. Xie, "Sharedmec: Sharing clouds to support user mobility in mobile edge computing," in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[16] C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum, "Optimizing the migration of virtual computers," *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 377–390, 2002.

[17] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, 2005, pp. 273–286.

[18] M. R. Hines, U. Deshpande, and K. Gopalan, "Post-copy live migration of virtual machines," *ACM SIGOPS operating systems review*, vol. 43, no. 3, pp. 14–26, 2009.

[19] V. Medina and J. M. García, "A survey of migration mechanisms of virtual machines," *ACM Computing Surveys (CSUR)*, vol. 46, no. 3, p. 30, 2014.

[20] J. Nider and M. Rapoport, "Cross-ISA container migration," in *Proceedings of the 9th ACM International on Systems and Storage Conference*. ACM, 2016, p. 24.

[21] R. Morabito, I. Farris, A. Iera, and T. Taleb, "Evaluating performance of containerized IoT services for clustered devices at the network edge," *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 1019–1030, 2017.

[22] G. T. 32.203, "3gpp ts 32.203, policy and charging control architecture."

[23] Nextepc. [Online]. Available: https://nextepc.org/