Orchestrating Heterogeneous MEC-based Applications for Connected Vehicles

Francesco Giannone[a,1], Pantelis A. Frangoudis[b,1,*], Adlen Ksentini[c], Luca Valcarenghi[a]

[a] *Scuola Superiore Sant'Anna, Pisa, Italy*
[b] *Distributed Systems Group, TU Wien, Vienna, Austria*
[c] *EURECOM, Sophia Antipolis, France*

Abstract

In the near future, 5G-connected vehicles will be able to exchange messages with each other, with the roadside infrastructure, with back-end servers, and with the Internet. They will do so with reduced latency, increased reliability, and large throughput under high mobility and user density. Different services with different requirements, such as Advanced Driving Assistance (ADA) and High Definition (HD) Video Streaming, will share the same physical resources, such as the wireless channel. Thus, a rigid orchestration among them becomes necessary to prioritize network resource allocation. This study proposes a *Connected Vehicle Service Orchestrator (CVSO)* which optimizes the Quality of Experience (QoE) of an in-vehicle infotainment video delivery service, while taking into account the required bandwidth for coexisting high priority services, such as ADA. To this end, we provide an Integer Linear Programming (ILP) formulation for the problem of optimally assigning a video streaming bitrate/quality per user to maximize the overall QoE, considering information from the video service and the Radio Access Network (RAN) levels. Our system takes advantage of recent developments in the area of Multi-access Edge Computing (MEC). In particular, we have implemented the CVSO and other service-level components and have deployed them on top of a standards-compliant MEC platform that we have developed. We exploit MEC-native services such as the Radio Network Information Service (RNIS) to offer the CVSO the necessary level of RAN awareness. Experiments on a full LTE network testbed featuring our MEC platform demonstrate the performance improvements our system brings in terms of video QoE. Furthermore, we propose and evaluate different algorithms to solve the ILP, which exhibit different trade-offs between solution quality and execution time.

Keywords: Multi-access Edge Computing, 4G/5G systems, Connected Vehicles, Video Streaming, Quality of Experience

1. Introduction

Future connected vehicles will exploit not only the information coming from their own sensors, as nowadays, but also those of other vehicles, other road users (i.e., pedestrians, bicyclists, trucks), and those shared by the road infrastructure to implement advanced safety services [1]. This trend poses significant challenges to the current communication system, as information must reach its destination reliably and in near-real time, beyond what the current wireless technology can offer.

5G, the next generation of mobile communications, promises improved performance in terms of reduced latency, increased reliability, and higher throughput under high mobility and user density [2]. The 5G Automotive Association (5GAA) categorizes a comprehensive list of connected vehicle applications in four main groups of use cases [3]: (i)

*Corresponding author
Email address: pantelis.frangoudis@tuwien.ac.at (Pantelis A. Frangoudis)

[1] Equal contribution.

Safety, (ii) Convenience, (iii) Advanced Driving Assistance (ADA), and (iv) Vulnerable Road User (VRU).

The applications grouped under the ADA use case (e.g., Real-Time Situational Awareness & High Definition Maps, See-Through and Cooperative Lane Change (CLC) of Automated Vehicles) are designed to improve traffic flow, traffic signal timing, routing, variable speed limits, and weather alerts. This group of applications is characterized by the most challenging requirements in terms of latency and requested capacity. Indeed, they require the distribution of an often large amount of data with high reliability and low latency.

At the same time, video streaming applications are gaining popularity and are becoming one of the major Internet services for mobile consumers. Thus, in-vehicle infotainment services are expected to stand out among the most popular connected vehicle services. A meticulous orchestration of such different services guaranteeing the requested Quality of Experience (QoE) is, therefore, mandatory.

Recent developments and ETSI-driven standardization activities in the area of Multi-access Edge Computing (MEC) [4] facilitate the cooperation of telecom operators and the automotive vertical industry and can play a fundamental role in this direction. MEC provides an execution environment for the deployment of third party applications at the mobile edge with a significant level of network awareness via standardized interfaces, such as the Radio Network Information (RNI) API [5]. This has the advantage of not only serving end users from edge servers, thus minimizing latency and reducing core network traffic, but also enabling the vertical service provider to perform network-aware optimizations exploiting real-time radio network information, such as network load and per-user channel quality indications.

This article presents the design, implementation, and evaluation of a *Connected Vehicle Service Orchestrator (CVSO)*, which aims to optimize the QoE of a video streaming service for infotainment, while guaranteeing the requested capacity to coexisting ADA services. This orchestrator is deployed as a MEC application, as is also the case for the two considered services. The orchestration algorithm involves the assignment of the appropriate video quality to each user to maximize the overall viewing experience; this problem is formulated as an Integer Linear Program (ILP). The orchestrator consists of a *Video Streaming Controller (VSC)* that implements and solves the ILP, and a *Radio Link Measurement (RLM)* component, which monitors the network status by accessing the RNIS, feeding such information to the ILP model. Based on the obtained results, the *VSC* sets the allowable video streaming bitrate, and thus quality, for each mobile user that the *Video Streaming Server (VSS)* application handles. Video is delivered using Dynamic Adaptive Streaming over HTTP (DASH) [6] technologies. This has the advantage of wide compatibility with standard HTTP servers and video players. Furthermore, we exploit specific features of the MPEG-DASH specification to control video quality in a MEC-assisted, RAN-aware manner, and *transparently* to video clients.

We evaluate our scheme through experiments over a fully functional LTE testbed based on OpenAirInterface (OAI),[2] which also includes our standards-compliant MEC platform implementation. Our results show that the CVSO brings significant improvements in terms of user experience compared with standard receiver-driven DASH video adaptation mechanisms and with non-adaptive streaming. Finally, we provide alternative algorithms to solve the ILP for video assignment, which we evaluate in terms of execution time and solution quality. In particular, we solve the problem to optimality using a state-of-the-art solver implementing a branch-and-cut algorithm and demonstrate that even for very large numbers of users, in the order of thousands, it takes less than 1.5 s to derive the optimal video representation assignment. This adds to our system's feasibility. We also implement a genetic algorithm

[2] `http://www.openairinterface.org/`

and a simple heuristic as low-cost alternatives. These algorithms allow the system operator to trade solution quality for faster execution time, which may be be important in scenarios with very large numbers of users and when very low response time is desired. Our results on optimality vs. running time can provide insight to the system operator on which strategy to select to solve the optimization problem under the given conditions (i.e., number of users simultaneously accessing the video service).

It should be noted that although a significant body of theoretical and experimental works [7–16] study various aspects of MEC-based video delivery, to the best of our knowledge, and other contributions aside, ours is the first to present a complete *standards-compliant* design and implementation, as well as testbed experiments of a RAN-aware MPEG-DASH video delivery architecture *and* at the same time over an ETSI MEC platform, with a view to the automotive vertical industry.

The rest of this article is organized as follows: In Section 2 we provide a review of the relevant literature. Section 3 is dedicated to the CVSO architecture from design to implementation, while our ILP model and algorithms for QoE-aware video bitrate assignment are presented in Section 4. Section 5 presents a user-centric (video QoE) and a provider-centric (ILP solution quality and execution time) performance evaluation of our scheme via testbed experiments and executions of video quality assignment algorithms in isolation, respectively. We discuss important operational decisions that we have not addressed in this article in Section 6 and conclude this work in Section 7. A list of acronyms used in the article can be found in Table 1.

Table 1: List of acronyms

AppD	Application Descriptor
ADA	Advanced Driving Assistant
CAM	Cooperative Awareness Message
CQI	Channel Quality Indicator
CVSO	Connected Vehicle Service Orchestrator
DASH	Dynamic Adaptive Streaming over HTTP
DENM	Decentralized Environmental Notification Message
eMBB	enhanced Mobile BroadBand
EPC	Evolved Packet Core
FQDN	Fully Qualified Domain Name
GA	Genetic Algorithm
HSS	Home Subscriber Server
ILP	Integer Linear Program
MME	Mobility Management Entity
MOS	Mean Opinion Score
MPD	Media Presentation Description
NSSMF	Network Slice Subnet Management Function
OVS	Open vSwitch
PI	Playout Interruption
PRB	Physical Resource Block
QoE	Quality of Experience
QP	Quantization Parameter
RAN	Radio Access Network
RLM	Radio Link Measurement
RNI(S)	Radio Network Information (Service)
S/P-GW	Service/PDN Gateway
SAND	Server And Network-Assisted DASH
UE	User Equipment
URLLC	Ultra Reliable and Low Latency Communication
VSC	Video Streaming Controller
VSS	Video Streaming Server

2. Related work

Edge computing, in general, and Multi-access Edge Computing, in particular, are seen as a critical enablers for applications across various vertical industries. This is reflected in a set of proposed use cases by ETSI [17], which include multimedia-related and automotive ones.

Apart from moving computation closer to the end device, which has direct advantages in terms of latency and reducing communication overhead in backbone links [18], MEC comes with a significant level of context awareness, mainly due to the availability of MEC-native services such as the RNIS and the Location Service. This allows deploying context-aware third-party applications at the mobile edge, which can take advantage of standardized interfaces to access RAN-level information at a fine granularity to optimize their operation. Also, this provides opportunities to operators to monetize on the value of their data by exposing them to third parties for profit. The volume and diversity of RAN-level monitoring data, and their correlation with application-perceived performance and with the network's operational state, raise the need and create opportunities for *RAN analytics* [19]. These can find use in a wide range of scenarios, from network troubleshooting and network resource management [20] to QoE-optimized service delivery, which is the focus of our work.

An early effort to use MEC for improved video delivery performance is due to Fajardo et al. [7]. At the time of their publication, the ETSI MEC architecture and the related services and interfaces were not yet mature. The authors introduce a network-assisted adaptation function as a MEC application and present simulations to demonstrate QoE improvements, focusing on H.264/SVC-encoded video.

Li et al. [11] present a scheme for joint access network selection and video quality assignment for DASH video delivery for multi-homed users. Their optimization algorithm runs as a MEC application and takes into account per-UE capacity information based on information from the RNIS. While they present technical details on how they take advantage of MEC in their video delivery architecture, they do not discuss how information from the RNIS is actually used, while their system is evaluated in a testbed where the RAN and MEC are emulated.

Sabella et al. [12], on the other hand, present an implementation of Radio Aware Video optimization in a fully Virtualized Network (RAVEN), one of ETSI MEC ISG's proofs of concept.[3] They implement a video streaming application on top of an LTE testbed, where mobile users capture and upload video which is consumed by other mobile users with the mediation of a video server. A video optimizer is notified of RAN congestion events or poor UE channel quality by the RNIS, and in turn adjusts the encoding parameters to adapt the video bitrate accordingly. Our RNIS implementation is different, but shares the same southbound API with OAI-based eNodeBs. Furthermore, our video streaming service is based on DASH and our adaptation logic is QoE-aware and executed transparently to video clients, while we also address management and orchestration issues.

Salsano et al. [8] implement a video streaming service in the context of the Superfluidity architecture [21], where the end-to-end video service, as well as MEC services such as traffic offloading are decomposed in a set of lightweight "reusable functional blocks," which can be seamlessly executed at the edge, the core, or remote clouds. Their focus is on demonstrating the architectural principles of Superfluidity, and their evaluation is around traffic offloading aspects and the impact of the transition of blocks between the core and the edge, rather than RAN aspects and ETSI MEC compatibility.

Taleb et al. [22] propose a video CDN slicing architecture compatible with ETSI NFV-MANO and ETSI MEC, combining empirical models of video QoE as a function of service workload with multi-level service monitoring (compute resource load, RAN-level conditions, video QoE probes running as VNFs, and QoE-related feedback provided by the clients) to drive resource allocation and elastic management mechanisms. The authors design decision making algorithms to identify the root cause of quality degradation and act accordingly via compute resource scaling or video adaptation. Instead of adapting the video stream via DASH techniques, as in our case, the authors propose

[3] `https://mecwiki.etsi.org/index.php?title=Ongoing_PoCs`

to modify it on the fly via a MEC transcoding service. Furthermore, their approach may require additional functionality at the video player to record and report video interruptions and other playout-related information, which is not the case in our design.

Mehrabi et al. [14] present a MEC-assisted DASH video delivery design, as well as low-complexity algorithms for client-to-edge-server assignment and bitrate selection. Via simulation, they show their algorithms to improve on achievable throughput, resource utilization, and QoE-related metrics such as the average video bitrate per client and initial buffering delay, while balancing with bitrate allocation fairness. We share the same motivation and approach, namely to provide network assistance in the streaming process. However, the authors rely on MPEG Server And Network Assisted DASH (SAND) [23], which involves explicit feedback from video clients and is not compatible with legacy DASH players. Instead of relying on SAND, we do so fully transparently to the client, showing in detail how ETSI MEC standard services and interfaces are used, and evaluating our implementation via testbed experiments over our ETSI-compliant MEC platform.

Ge et al. [10] focus on user-generated high-definition video content, and propose to selectively cache video segments at the edge for a limited amount of time. While they also carry out manifest manipulations at the edge (albeit on HLS-formatted video), they focus on 4K live video streaming scenarios, where quality adaptations should be limited or disabled, or are not possible at all. In the case of video on demand, Ge et al. [9] propose techniques for adaptive prefetching of DASH video chunks at the edge. In both cases, the authors present extensive experimental results over an LTE-A testbed, but not using ETSI MEC as their edge computing platform. Tan et al. [13] also implement an edge caching system where the segment representations to cache are selected based on information about content popularity and the signal conditions per UE.

In the context of immersive video applications, Shi et al. [15] design and implement MEC-VR, an end-to-end Virtual Reality system which delivers 360°video, where rendering takes place with the collaboration of a remote server and the client device. In contrast with our work, MEC-VR is latency-centric. The authors use edge computing to reduce remote rendering delays, and the amount of video that is rendered remotely is decided according to latency. In addition, they do not consider any MEC-specific features and services, other than placing computation close to end devices. Rigazzi et al. [16] present a three-tier architecture spanning the cloud-fog-edge continuum, with compute intensive video processing tasks being offloaded to the edge where GPU-powered devices reside. Service orchestration is implemented using the Fog05 framework.[4] That work focuses on some complementary aspects to ours, namely computation offloading and end-to-end orchestration, for a different type of video streaming application. Also, an ETSI MEC compatible architecture is not directly assumed and the services thereof, such as the RNIS, are not exploited, while the evaluation is carried out for a single-user scenario.

The role of MEC in the automotive space has been highlighted by Giust et al. [24] and ETSI has recently published a group report on MEC support for V2X use cases [25]. A safety-critical use case is intersection control and collision avoidance, where the role of MEC in extending vehicles' perception range is studied by Olmos et al. [26]. This is also the subject of the work of Avino et al. [27], who design and implement a MEC-based extended virtual sensing service that fully complies with ETSI MEC and ITS [28] specifications, including a collision avoidance algorithm. Notably, this service is deployed over our MEC platform, and, as such, it is the first of its kind to be implemented and evaluated on top of a real, standards-compliant, MEC system.

In the context of advanced driving assistance, Keivani et al. [29] propose a vision-based system for forward colli-

[4] `https://fog05.io`

sion warning, using smartphones onboard to capture and transmit video to edge servers, where compute-intensive detection and tracking algorithms are executed. Maheshwari et al. [30] present EdgeDrive, a augmented reality ADA system using head-mounted devices, offloading computation tasks to containers that can migrate across edge hosts. These systems focus on a different family of vehicular applications to ours, are not tailored to ETSI MEC architectures, and are evaluated via simulation or emulation.

Nguyen at al. [31], on the other hand, focus on the protection of vulnerable road users and propose a system where pedestrian or other VRU context information collection, as well as the execution of collision detection algorithms, are either carried out on user devices or are adaptively offloaded to MEC servers. Napolitano et al. [32] implement a VRU application on Android devices, whose server-side components are executed at MEC hosts, and compare different access technologies (LTE vs. Wi-Fi) and placement strategies (MEC vs. cloud) in terms of latency to deliver Cooperative Awareness Messages (CAM) to the server.

Service migration across MEC hosts, an issue that we are not addressing in this work, is the focus of Campolo et al. [33]. They propose a reference architecture integrating ETSI MEC and Cellular V2X, use pre-relocation techniques to migrate service components to the appropriate MEC host using knowledge or predictions of the trajectories of vehicles, and experimentally demonstrate improvements in terms of service disruption times using a docker-based proof of concept.

A summary and classification of the relevant state of the art is presented in Table 2.

3. Connected Vehicle Service Orchestrator Architecture and Components

This section details the Connected Vehicle Service Orchestrator architecture and its MEC-based components, namely the Video Streaming Controller and the Radio Link Measurement module. In addition, the Video Streaming Server application and its interaction with clients are described.

3.1. Managing coexisting automotive service components

Figure 1 shows the proposed application orchestration scheme. The two orchestrator components, namely the *VSC* and the *RLM*, and the *VSS* are deployed at a MEC host. The ADA component (road safety application) is also depicted. As elaborated next, its capacity requirements are known to the CVSO, and are taken into account by considering an amount of necessary radio link resources as reserved per connected vehicle. The internal workings of this component are outside the scope of this work. The interested reader may refer to [27] for more details on an ADA service for collision avoidance deployed over our MEC platform.

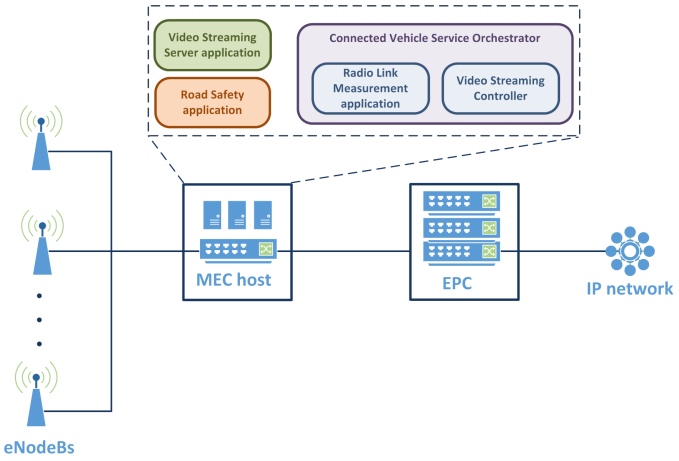


Figure 1: Proposed Architecture.

In the *network-slicing* enabled 4G/5G network environment that we target [34], high-priority automotive services such as the road safety application, can be served via end-to-end network slices with dedicated network and compute resources. Slice orchestration components are in charge of allocating RAN, transport and core network resources for slices as per the request of the automotive vertical service provider. These allocation decisions are enforced by

Table 2: Summary of the relevant (mobile/multi-access) edge computing literature. ✓: contribution on the topic, feature supported or aspect addressed; (✓): aspect partially addressed or feature partially supported; n/a: not applicable; -: not supported/addressed.

		Fajardo et al. [7]	Li et al. [11]	Sabella et al. [12]	Salsano et al. [8]	Taleb et al. [22]	Mehrabi et al. [14]	Ge et al. [9]	Ge et al. [10]	Tan et al. [13]	Shi et al. [15]	Rigazzi et al. [16]	Olmos et al. [26]	Avino et al. [27]	Keivani et al. [29]	Maheshwari et al. [30]	Nguyen et al. [31]	Napolitano et al. [32]	Campolo et al. [33]	Our work
Video service	DASH/HLS	✓	✓	-	✓	-	✓	✓	✓	✓	-	✓	n/a	n/a	-[c]	-[d]	n/a	n/a	n/a	✓
	Network-assisted	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	-	n/a	n/a	-	-	n/a	n/a	n/a	✓
	Bitrate/quality adaptation mechanisms	✓	✓	✓	-	✓	✓	-	-	-	-	-	n/a	n/a	-	-	n/a	n/a	n/a	✓
	Legacy DASH/HLS video clients	(✓)[a]	✓	-	✓	-	-	✓	✓	✓	n/a	(✓)[b]	n/a	n/a	n/a	n/a	n/a	n/a	n/a	✓
	QoE-driven modeling & optimization	(✓)	✓	-	-	-	✓	-	-	(✓)	-	-	n/a	n/a	n/a	-	n/a	n/a	n/a	✓
Architecture/design features	MEC standards-compliant	-	✓	✓	(✓)	-	-	(✓)	(✓)	(✓)	-	-	-	✓	-	-	-	✓	✓	✓
	4G/5G cellular	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	(✓)	✓	✓	-	✓	✓	✓	✓
	RAN aware	✓	✓	✓	-	✓	✓	✓	✓	✓	-	-	-	-	-	-	-	-	-	✓
	Management/ orchestration aspects	-	-	-	✓	✓	-	-	-	-	-	✓	-	✓	-	-	-	-	✓	✓
	Vehicular environment	-	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓	✓	✓
Testbed	Real 4G testbed	-	-	✓	✓	-	-	✓	✓	✓	✓	-	-	✓	-	-	-	✓	-	✓
	ETSI MEC platform	-	-	✓	-	-	-	-	-	-	-	-	-	✓	-	-	-	-	-	✓

[a] Relies on MPEG SAND.

[b] Focuses on 360 video. Requires the player to report the user orientation.

[c] Uplink video transmitted from cameras to edge servers for processing.

[d] Augmented reality ADAS applications. Video from head-mounted devices is sent to edge servers for processing.

Network Slice Subnet Management Functions (NSSMF) which arbitrate low-level resource allocation across competing slices by communicating with control-plane entities such as RAN controllers, as we have described in our prior work [35]. The automotive vertical typically requests the deployment of network slices tailored to the QoS requirements of each service component, which are highly diverse. In our case, the road safety service is considered a high-priority one, with typically low bandwidth but high reliability and low latency requirements, and can thus be considered to belong to the Ultra Reliable and Low Latency Communication (URLLC) service class. The infotainment service, on the other hand, has higher bandwidth demands but may be delivered in a best-effort manner within a low-priority enhanced Mobile BroadBand (eMBB) slice.

Note that the CVSO is managed by the automotive vertical, operating independently of slice orchestration and management, which is implemented by the network operator. Therefore, while the RAN resources and, in turn, the QoS required by the road safety service can be guaranteed by a high-priority URLLC slice, it is up to the CVSO to optimize the performance of the video streaming service via *intra-slice* coordination of competing video flows. One of the CVSO's tasks in this process is to estimate what is the available bandwidth left for the video streaming service in terms of *Physical Resource Blocks (PRBs)* per User Equipment (UE), and make the best of it by appropriately and dynamically selecting the video bitrate assigned to each user. This available bandwidth depends on (i) the throughput requirements of the road safety service and (ii) the PRBs that the slicing system dedicates to achieve this. The latter may fluctuate with the changes in the channel quality of UEs: the lower the channel quality, the higher the number of radio resources that need to be allocated to attain the same throughput performance.

The class of road safety applications that we assume in this work involves the delivery of Decentralized Environmental Notification Messages (DENM) from the network/infrastructure to vehicles, as specified in ETSI EN 302 637-3 [36]. These messages may be generated periodically or asynchronously in an event-driven manner,[5] and thus may involve constant- or variable-bitrate downlink traffic per vehicle. Typically, DENM packets are small, in the order of tens of bytes, and their transmission frequency is low. For example, in the experimental evaluation of Santa et al. [37], one DENM packet per second is generated with a size of 49 bytes. According to ETSI [38], DENM transmission frequency depends on the nature of the application, usually ranging from 1 to 10 Hz, and can be adjusted depending on the criticality of the situation. These can provide sensible upper bounds on the bandwidth that needs to be reserved by the network slice within which the road safety service is delivered. The traffic characteristics of the road safety service are known to the CVSO. This information is combined with real-time per-UE channel quality data readily available by the RNIS in order to estimate the RAN resources (PRBs) that are not reserved for the safety service and are thus usable by the video streaming one. This takes place by mapping the UE channel quality to an achievable throughput.

3.2. Video streaming control and RAN monitoring

The *VSC* makes the orchestration decisions. Thus, it performs the following actions:

1. It periodically builds an ILP problem to assign video representations/qualities to each connected user accessing the video service, taking into account the Channel Quality Indicator (CQI) values received from the *RLM*.
2. It solves the above optimization problem.
3. According to the solution of the optimization problem, it selects the appropriate video representation for each user and it communicates it to the VSS as described in Section 3.3.

[5] After an event occurs, the DENM may be repeated periodically with an application-defined frequency.

Note that the VSC needs to be aware of the subset of the connected UEs that are simultaneously participating in the video streaming service.

The purpose of the *RLM* is to maintain a view of the status of the RAN conditions. In particular, it accesses the RNIS over its standardized interface to have an updated view of the number of users per eNodeB in the area that it manages[6] and their CQI values. This information is critical for estimating each user's link capacity, and, in turn, the video bitrate that it can sustain.

The CQI values are normally reported by the UE to the eNodeB via standard LTE procedures. In our MEC platform implementation, which is tailored to OAI, we use the FlexRAN architecture and protocol [39] to extract them (as well as a wealth of other RAN-level information) from the eNodeB and make them available to the RNIS, so that MEC applications such as the RLM can consume them.[7] The *VSC* receives CQI updates from the *RLM* and uses this information as ILP input.

3.3. MEC-assisted video streaming service

The *VSS* contains the video available for streaming and the information describing the content. In our proposed scheme, the *VSS* is also deployed as a MEC application instance and plays the role of an edge video cache. This has the advantage of serving users from a nearby location, thus reducing startup latencies and saving on backhaul network resources. Our adopted technology for video streaming is Dynamic Adaptive Streaming over HTTP (DASH). DASH is a streaming technique that enables high quality streaming of media content over the Internet delivered from conventional HTTP web servers. MPEG-DASH works by partitioning the content into a sequence of small video segments, each containing a short duration of video (typically a few seconds).

Information about the available representations needed by clients is stored in the Media Presentation Description (MPD) file [41]. The MPD file structure follows a hierarchical data model containing one or more *periods*. Each period contains video characteristics such as available bitrates, resolutions, codecs, and segment IDs.

Normally, to play the content, an MPEG-DASH client first obtains the MPD over HTTP. By analyzing the MPD, the client learns about all the content characteristics. Then, it selects the appropriate representation among the available listed in the MPD according to the network conditions. Finally, it starts streaming the content by fetching the segments related to the selected representation by issuing HTTP GET requests. After appropriate buffering to deal with network throughput variations, the client continues fetching the subsequent segments of the selected representations. Meanwhile, the client may monitor the available network bandwidth and its fluctuations and appropriately adapt to it by fetching segments of a different representation characterized by a lower or higher bitrate, as listed in the MPD, to match the current network conditions.

One disadvantage of pure receiver-driven approaches for bitrate adaptation is the fact that the video client takes adaptation decisions based on bandwidth estimates only, lacking accurate knowledge of the actual network conditions. This is one of the issues that the MPEG SAND standard [23, 42] aims to address. SAND involves the exchange of quality-related information between SAND-aware clients and servers and between DASH-Aware Network Elements (DANE) for improved adaptation decisions. SAND, however, requires specific extensions to DASH clients to support it.

Our approach to offer server assistance in the video

[6] Each edge data center and the respective MEC platform may correspond to a region containing multiple mobile network cells. In this scenario, each CVSO would orchestrate the video delivery and ADA services for all the users/vehicles within these cells. For simplicity, we can consider that there is a single MEC Platform, and thus, CVSO instance, per network cell.

[7] For more details on our MEC platform and RNIS implementation, the reader is referred to [40].

delivery process is different, since one of our targets is for our server-side mechanisms to be transparent to the clients and to support default DASH players without any modifications. However, we plan to study how to integrate SAND mechanisms in our design in future work.

In our proposed scheme, since the VSS is deployed at a MEC host, it can take advantage of the availability of RAN-level information coming from the RNIS and adapt the contents of the MPD files requested dynamically and on a per-user basis, according to the current link capacities of the involved clients. We have implemented this functionality as follows.

The VSS operates an HTTP front-end proxy which receives user requests for MPD files. This proxy also has an internal REST HTTP interface where the VSC posts updates about the outcome of the video representation assignment algorithm. In particular, each time the VSC is executed, it updates the VSS front-end with <user IP address-video representation> pairs. When the VSS front-end receives a request for an MPD file, it fetches it from the VSS, modifies it so that it contains only the appropriate video representation, and sends it back to the user. This procedure is transparent to the video client, which proceeds with downloading the video chunks that correspond to the representation selected for it by the VSC via the URLs specified in the MPD file.

To force the clients to periodically request an updated MPD that reflects the current network state, we use a mechanism that is specified in MPEG-DASH, thus maintaining the compliance of our solution with the standard: we exploit the `minimumUpdatePeriod` MPD attribute. When this attribute is set in the MPD file, the client re-fetches a fresh version of it when the timeout defined in the `minimumUpdatePeriod` expires.

Finally, to ensure that the VSC always has an accurate view of the subset of connected users in the cell which are *actually* consuming the video service and thus should be considered by the algorithm, we rely on notifications from the VSS: Each time the VSS front end receives an MPD request, it posts an update to the VSC with the client's IP address, reporting it as active. If a client has not requested a file for longer than a predefined time interval (adequately larger than `minimumUpdatePeriod`), it is considered inactive and is removed from the VSC's list of video service participants.

3.4. Service preparation and deployment

To deploy an instance of our service, a standardized procedure takes place, where the service provider (automotive vertical) packages, onboards, and requests the instantiation of its application components, namely the RLM, VSC, and VSS, over the reference points specified by ETSI MEC. In particular, the vertical accesses the OSS/BSS via a *Customer Facing Service (CFS)*, which forwards application lifecycle management operations (e.g., onboarding, instantiation, termination) to the OSS/BSS over the Mx1 reference point. We should note that Mx1 is not further specified [4, Clause 7.2.3]. In our implementation, we have built it as a web portal which controls authentication and authorization of third parties deploying MEC applications, and directly interfaces with the MEC Orchestrator (MEO) by using the latter's Mm1 REST API endpoint, as specified in ETSI MEC 010-2 [43]. The ETSI MEC specification also supports two alternative instantiation mechanisms. For specific applications, the MEC operator may request their instantiation directly to the MEO over Mm1. This however applies mainly to operator-controlled services, rather than third-party vertical services as in our case. It is also possible to create a MEC application instance by communicating with a special entity, the UE Application LifeCycle-Management Proxy, if the latter is provided by the MEC system. This takes place over the Mx2 reference point, as specified in ETSI MEC 016 [44] and elaborated by Sabella et al. [45].

Each MEC application package is described by an *Application Descriptor* (AppD). The format of the AppD is

specified in ETSI MEC 010-2 and it is the equivalent of a VNF descriptor (VNFD) in ETSI NFV-MANO [46], including a path to the application image, requirements for specific MEC services, application latency and minimum compute resource constraints.

In line with our microservices-inspired approach, each component of our service is prepared as a separate MEC *application package*, and an issue to be addressed is how these components will discover each other once they are instantiated. In our design, the vertical uses the special `appDNSRule` field of the AppD to provide one (or more) Fully Qualified Domain Name(s) (FQDN) for each specific component. These DNS rules are applied by the MEC platform at application instantiation time by updating the MEC DNS service with the respective records. They are then used by the RLM, VSC, and VSS to discover each other and also by the UEs to be able to connect to the VSS and discover other application endpoints that the service provider may expose to its end devices. It is reasonable to assume that in this automotive context, in-vehicle devices are preconfigured to access the aforementioned services at well-known FQDNs and URLs; it is then a matter of the MEC system to resolve these names to the IP addresses of the appropriate service instances.

It is also important that the applications are architected with MEC awareness. This implies that they should be able to discover the MEC platform and consume its services via well defined interfaces. The latter takes place over the Mp1 reference point, with procedures and interfaces standardized in ETSI MEC 011 [47]. However, the standard does not define how applications discover the MEC platform itself in the first place. Again, we implement this functionality via DNS, preconfiguring MEC application packages with a well-known FQDN for the MEC platform.

4. QoE-aware video representation selection: Problem formulation and algorithms

In our application scenario, the video service provider makes videos available in a distinct set of representations, each with its own bitrate which maps to a specific QoE value. It should be noted that for the same codec settings, bitrate, and resolution, and even if we ignore the effects of other impairments (e.g., buffering events, display qualities), different videos might come with different QoE since the content itself (e.g., the amount of motion in the video) affects the relationship between video quality and bitrate. In this work, we do not take into account such effects.

The purpose of the VSC is to select the most appropriate video representation for each user, considering the available network capacity, load (i.e., number of users sharing radio resources), and the individual link characteristics (channel quality), to maximize the overall user experience expressed as the sum of the users' QoE values. We formulate this problem as an ILP with the following parameters:

- n: number of mobile users connected to the *VSS*;
- m: number of video representations available in the *VSS*;
- b_j: video bitrate of representation j;
- E_j: QoE corresponding to video representation j;
- R^i_{1PRB}: achievable bitrate by user i in one Physical Resource Block (PRB); it is extracted from *Table 7.1.7.1-1* of [48] and it is a function of the CQI value of the i^{th} user;
- R^i_{MAX}: maximum achievable bitrate by user i according to its CQI value.[8] It is similarly extracted

[8] This value is computed by assuming that user i is assigned all the PRBs available for the video streaming service. This also captures the bandwidth requirements of the ADA service per individual user, which has higher priority. A per user bandwidth requirement for ADA can be translated to PRBs for a given UE CQI; R^i_{MAX} is then calculated based on the *remaining* PRBs.

from *Table 7.1.7.1-1* of [48];

- PRB^{VSS}: portion of the overall number of PRBs usable by the video streaming service.

The binary variable x_{ij} indicates whether mobile user i is assigned video representation j. In our problem formulation, QoE is expressed in terms of the Mean Opinion Score (MOS). The MOS is defined as the expected rating that a panel of users would give to the quality of the transmitted video in the 1-5 scale, where 1 represents the poorest quality and 5 an excellent one [49]. QoE can be estimated from video quality parameters, such as interruption statistics and encoding parameters, which can be translated to MOS as we explain in Section 5. The estimated MOS that corresponds to each representation (E_j values), which is related to its bitrate and, in turn, to the specific encoder parameters used to produce it, is calculated offline and is reasonably assumed to be known to the VSC. The problem is therefore modelled as the following binary ILP.

$$\text{maximize} \quad \sum_{i=1}^{n}\sum_{j=1}^{m} E_j x_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{b_j}{R^i_{1PRB}} x_{ij} \leq PRB^{VSS} \tag{2}$$

$$b_j x_{ij} \leq R^i_{MAX}, \forall i \in [1,n] \tag{3}$$

$$\sum_{j=1}^{m} x_{ij} \leq 1, \forall i \in [1,n] \tag{4}$$

$$x_{ij} = \{0,1\}, \forall i \in [1,n], j \in [1,m] \tag{5}$$

Constraint (2) ensures that the sum of all the PRBs necessary for the video representations assigned to mobile users is less than those available to the video streaming service. In other words, this guarantees that the total required bandwidth for the video service does not exceed the capacity. Constraint (3) ensures that a video representation is not assigned to user i if its bitrate cannot be accommodated by the user (individual user capacity constraint). Finally, constraint (4) guarantees that a user is assigned at most one video representation and (5) restricts the decision variables to binary values.

Note that due to constraint (4), this problem always admits a (trivial) feasible solution, where no user is assigned any video. This could be the case in extreme conditions where there is not enough capacity to admit even a single user, given the bandwidth requirements of the available video representations.

Proposition 1. *The QoE-aware video representation selection problem is NP-hard.*

Proof. The above problem is a *generalization* of the k-item Integer Knapsack Problem (kIKP), also known as the Cardinality Constrained Knapsack Problem (see [50, Chapter 9.7] for a study of its binary version), where a knapsack of capacity c and m types of items are available, each of the latter associated with a profit p_j and a weight w_j, with $j = 1, \ldots, m$. The objective is to select a number (integer) $y_j \geq 0$ of items of each type to put in the knapsack such that their sum of profits $\sum_{j=1}^{m} p_j y_j$ is maximized, subject to the constraints that $\sum_{j=1}^{m} w_j y_j \leq c$ (capacity constraint) and $\sum_{j=1}^{m} y_j \leq k$ (cardinality constraint; no more than k items can be placed in the knapsack). In particular, by assuming that all users enjoy perfect channel conditions and are thus indistinguishable, with identical values $R^i_{1PRB} = r$ in constraint (2), and *restricting* individual link capacities to a value $R^i_{MAX} = M \geq PRB^{VSS} r$ for all $i = 1, \ldots, n$ (i.e., higher than the overall network capacity, thus eliminating individual capacity constraints), we have an equivalent kIKP instance, where item type j corresponds to video representation j with $p_j = E_j$ and $w_j = b_j$, knapsack capacity equals the total bandwidth allocated to the video streaming service ($c = PRB^{VSS} r$), and $k = n$. Because of constraint (4), no more than k representations (items) can be selected. Since there are no individual link capacity constraints and users are identical and indistinguishable, they do not affect the assignment process and any assignment that maximizes (1) directly

corresponds to an optimal kIKP item selection and *vice versa*. The QoE-aware video representation selection problem is thus NP-hard, as a generalization of kIKP (itself a generalization of the NP-hard Integer Knapsack Problem). ■

Despite the NP-hardness of the problem, our experimental results presented in Section 5.3 show that it can be solved to optimality in acceptable time for reasonably sized problem instances using state-of-the-art solvers, and in particular via the branch-and-cut algorithm [51] provided by the CPLEX optimization studio. We have also designed and implement the following two heuristics, which solve the problem considerably faster as the number of UEs grows, albeit at the cost of deriving suboptimal solutions.

Genetic algorithm. A genetic algorithm [52] represents problem solutions as *chromosomes*, each composed of a group of genes. In our case, a chromosome encodes the set of UEs and a gene is the video representation assigned to a single UE. The algorithm maintains a solution pool with S candidates, and explores the solution space by iteratively applying a sequence of *crossover* and *mutation* operations on selected chromosomes. At each iteration of the algorithm, i.e., a *generation*, pairs of chromosomes are selected uniformly at random for crossover and exchange genes, thus creating new offspring, i.e., new candidate solutions. Also, each chromosome mutates with a very low probability. In this case, one of its genes is selected uniformly at random and its assigned video representation is changed randomly. The purpose of mutation is to introduce diversity in the solution pool. Whenever a new candidate solution is generated either by mutation or by crossover, all the problem constraints are checked and, if violated, the candidate is not added to the solution pool. At the end of a generation, the top-S solutions in terms of a fitness function (in our case, the objective function (1)) form the new solution pool. The algorithm terminates if the fitness function value of the best solution in the pool remains unchanged (or changes by less than a very small threshold value) for a fixed number of generations or a if maximum number of generations is reached. The chromosome with the highest fitness value is the solution returned by the algorithm. The running time of the algorithm depends on its parameters, i.e., the solution pool size, the number of generations, and the crossover and mutation rates, that is the number of crossover operations that take place at each round and the probability that a chromosome is subject to mutation, respectively.

Baseline. This is a simple iterative algorithm that first sorts UEs and video representations in decreasing CQI and increasing bitrate order, respectively. It begins by assigning the lowest-bitrate representation to as many UEs as possible, and iterates over representations operating in the same manner, aiming to assign better bitrate (and thus quality) representations to the UEs which have the capacity to sustain them. The algorithm terminates when there is no more available network capacity or when there are no users whose video bitrate can be improved due to their individual link constraints.

5. Performance Evaluation

5.1. Testbed description and experimental methodology

The testbed shown in Figure 2 is utilized for our experimental evaluation. We have implemented a complete MEC system tailored to OAI that complies with ETSI MEC specifications, also including a fully fledged RNIS. Our testbed includes an OAI LTE network, with the appropriate extensions to interface with our MEC platform.

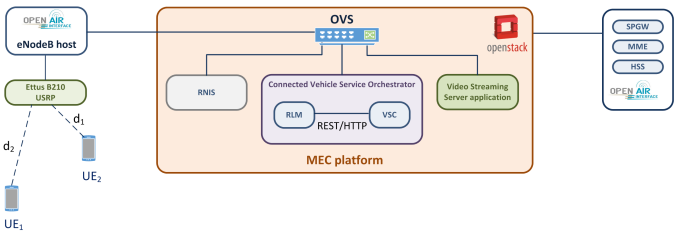


Figure 2: Utilised testbed.

The OAI-based Evolved Packet Core (EPC) contains the implementation of the following core network elements: the Serving and PDN Gateways (S/P-GW) bundled in a single software component, the Mobility Management Entity (MME) and the Home Subscriber Server (HSS). They are deployed as a set of virtual machines on top of the kvm hypervisor. At the S/P-GW level, we have followed the Control-User Plane Separation principle [53], and use it to implement traffic steering towards MEC applications. In particular, the S/P-GW control and data plane are split, and user data plane traffic is handled by an Open vSwitch (OVS) instance. When specific MEC applications are deployed to which traffic needs to be offloaded (as is the case, e.g., for the VSS), the appropriate traffic redirection rules are installed by the MEC platform to OVS and the latter steers traffic appropriately. Further details on our MEC platform are not within the scope of this article.

The OAI eNodeB is a standards-compliant implementation of a subset of LTE Release 14. The Ettus B210 USRP device is a fully integrated, single-board, Universal Software Radio Peripheral platform and acts as radio front-end, which is connected to a dedicated Intel i7 host running the OAI eNodeB software.

Two Huawei EE372 dongles, each connected to a laptop, equipped with a preconfigured SIM card able to connect to the OAI mobile network act as UEs. The UEs are connected to the same eNodeB and are located at different distances from the eNodeB, in order to make sure that they have significantly different CQI values. A first UE (i.e., UE_1) is located at a distance d_1 while a second UE (i.e., UE_2) is placed at a distance d_2, as shown in the figure.

The MEC platform hosts the *CVSO* and its MEC-based components, namely the *VSC* and the *RLM*, as well as the *VSS*. The RLM consumes CQI reports from the RNIS and makes them available to the VSC to run the video representation assignment algorithm. The VSC communicates with the VSS front-end to notify it of up-to-date video representation assignments over the latter's internal interface via HTTP (see Section 3.3).

In our experiments, we used different representations of the same test video (the 10m54s Blender Foundation's "Elephants Dream" movie[9]), which are stored in the *VSS* and are made available for streaming. Using the ffmpeg tool,[10] the test clip is H.264/AVC-encoded in six files with different bitrates approximately equal to 256 $Kbps$, 512 $Kbps$, 1024 $Kbps$, 2048 $Kbps$, 4096 $Kbps$, and 8192 $Kbps$, respectively, with a 1280×720 resolution.

The different representations are then prepared for DASH delivery using GPAC's MP4Box utility.[11] Each segment has a duration of $2s$.[12] The `minimumUpdatePeriod` MPD file attribute is set so that the mobile users request an updated version of the MPD file every 10 s.

The proposed *CVSO* is then evaluated in terms of the achieved QoE when the movie is streamed simultaneously to both UEs. QoE is principally affected by the video encoding quality, determined by the Quantization Parameter (QP) used for encoding, and the Playout Interruptions (PI) caused by buffering delays due to insufficient bandwidth. We make the assumption that the effect of the impairments related to interruptions and picture quality (QP) on QoE is additive.[13] This is a typical assumption for QoE assessment, both for video [56, 57] and VoIP services [58]. In order to measure these impairments, we use a combination of existing tools as in [11].

Regarding the effects of interruptions (I_{PI}), we use

[9] `https://orange.blender.org/`

[10] `https://trac.ffmpeg.org/`

[11] `https://gpac.wp.imt.fr/`

[12] According to Lederer [54], this is an appropriate segment duration for an environment with low delay (due to MEC-based delivery) but high bandwidth fluctuations (due to the potentially varying channel conditions of mobile users), helping clients adapt faster to bandwidth changes and limit stalls.

[13] The theory that quality impairment factors have an additive effect on the psychological scale is due to Allnatt [55].

the Pseudo-Subjective Quality Assessment (PSQA)[14] approach [59], and in particular the PSQA tool of Singh et al. [60], configuring it to ignore QP-related impairments. On the other hand, for measuring picture quality impairments (I_{QP}) we apply the QoE estimation model developed in the context of the VIPEER project [61]. This tool ignores the effects of interruptions and only considers the average value of the QP for each QoE measurement window (16 s of video, in our case). Subtracting the I_{QP} and I_{PI} values from the maximum possible MOS corresponding to excellent quality (i.e., 5), we get the MOS value for the specific measurement window:

$$MOS = 5 - I_{QP} - I_{PI}. \quad (6)$$

To be able to measure and export the PI and QP information necessary for our QoE calculations, we have appropriately extended the open-source MP4Client video player and ffmpeg's libavcodec encoding/decoding library, and installed them at the UE hosts.

5.2. Quality of Experience evaluation

In this section, we present the results of our experimental evaluation, showing our proposed scheme to offer improvements in terms of user experience. For the sake of comparison, we tested four different video delivery scenarios.

1. In the first scenario, we apply our proposed MEC-assisted mechanisms and video representation assignment algorithm executed by the *CVSO* aiming to maximize the overall user experience. The experimental results obtained under this scenario are labelled as "RAN-aware" and are shown in green boxes in Figure 3 and Figure 4.
2. In the second scenario, the video bitrate is fixed to a low value for all users. The corresponding curves in Figure 3 and Figure 4 are labelled as "Fixed [low quality]" and are shown with blue cross points.
3. In the third scenario, the video bitrate is fixed to a medium value for all users. Such experimental results labelled as "Fixed [medium quality]" are depicted by means of pink dots in Figure 3 and Figure 4.
4. In the last scenario, the default receiver-driven bitrate adaptation algorithm is used, where the video player estimates the available bandwidth and accordingly selects one of the available video representations. The experimental results obtained under this scenario are labelled as "Receiver-driven" and are depicted as orange stars in Figure 3 and Figure 4.

In all our experiments the video streams of the two UEs are started with a maximum difference of 5 s, thus their playout is not synchronized. We ignore the first 30 s of playout, considering it a warm-up period and so as to ensure that both video streams are already ongoing when we start our measurements. We should note that x-axis corresponds to *video time* and not wall clock time. This is necessary for a fairer comparison of the achieved MOS across the different benchmarks: Each QoE value corresponds to a window of 16 s[15] and is a function of the video interruptions and the average QP value during this window. The QP value is controlled by the encoder, depends also on the video *content* characteristics, and fluctuates during the video. Therefore, we always compare QoE values that correspond to the same 16 s window and are thus calculated over the same sequence of video frames, across the different experiments.

In scenario 1, the *VSC* is aware of the CQI values of each mobile user accessing the video service and is able to

[14] The PSQA methodology involves training a Random Neural Network (RNN) using data from subjective tests. The trained RNN classifier can then be applied to calculate the expected MOS for specific values of the input parameters.

[15] Since we ignore the first 30 s, the first MOS value corresponds to the 46-th second of video time.

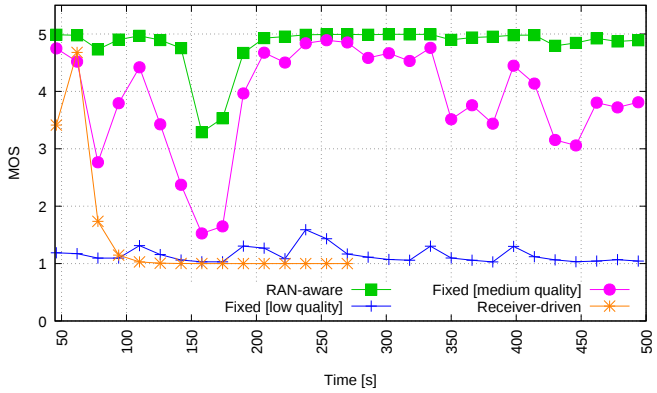


Figure 3: QoE for UE_1 (CQI=15).

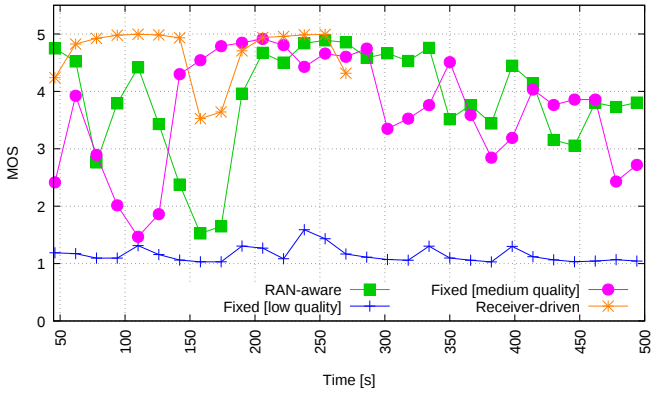


Figure 4: QoE for UE_2 (CQI=10).

optimize the MOS of UE_1 for the entire video duration obtaining the highest MOS values as shown in Figure 3. Regarding UE_2, as shown in Figure 4, the performance is lower due to a lower CQI value.

In the second and third scenario, there is no awareness of the network conditions and the video streaming bitrate is fixed. In the second scenario, a low video quality representation corresponding to a low video streaming bitrate is imposed to satisfy also the video streaming demand of the most disadvantaged UE (i.e., UE_2). Due to the low video streaming bitrate adopted in this scenario, the MOS experienced by both mobile users is less than the one experienced in the first scenario where the video quality assignment is optimized. Then, in the third scenario, to increase at least the MOS of the most advantaged UE, video quality is improved by fixing a higher video streaming bitrate. The user experience of both UEs is improved, as depicted in Figure 3 and Figure 4, but, in particular for UE_1, it is lower than the MOS obtained in scenario 1, where the appropriate video representation is assigned with RAN awareness.

In the fourth scenario, the adaptation of the video bitrate is left to the DASH client. In our tests, we observed that the mobile user which starts streaming first (e.g., UE_2) quickly switches to a high-bitrate representation, thus requesting most of the available RAN capacity and maximizing its MOS. This causes the user that joins afterwards to start with a low initial estimate of the available bandwidth, thus selecting the lowest-quality representation and experiencing a low MOS, as shown in Figure 3 and Figure 4. It should be noted that the receiver-driven curves in both figures are truncated, since UE_1's video stalls indefinitely after a few seconds. We repeated the experiment multiple times, and this is something we consistently observed across all repetitions. We attribute this to a combination of the OAI eNodeB scheduler's behavior in the face of downlink congestion[16] and the behavior of the specific video player when the TCP connections with the VSS experience very large delays.

Finally, we calculate the average of the MOS values of UE_1 and UE_2 for each of the above scenarios. As shown in Figure 5, the case when the *CVSO* is used scores consistently higher.

5.3. ILP solution performance

Turning our attention to the performance of the *CVSO* with respect to solving the ILP for video representation as-

[16]The OAI RAN in our testbed operates in FDD mode with a 5 MHz channel bandwidth (25 PRBs). In these settings, we measured the maximum downlink TCP throughput when a single UE was connected to the eNodeB with a very good channel quality ($CQI = 15$) to be in the order of 16 Mbps. The video bitrate of the highest quality representation in our experiments is already higher than 8 Mbps, not including the overheads associated with video segmentation and the HTTP protocol that is used as the underlying transport.

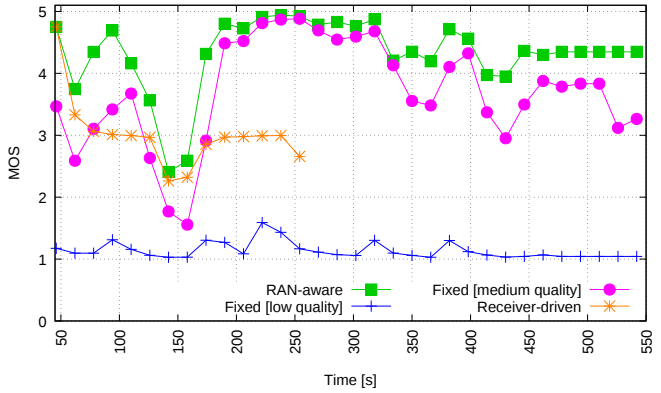


Figure 5: Average QoE Comparison.

signment, we perform a set of experiments where we generate problem instances of varying sizes and evaluate the execution time and the quality of the derived solutions for the three candidate algorithms[17] presented in Section 4. We generated six video representations and measured their average MOS[18] using the tools described in Section 5.1. We ran our tests on an Intel i7 host with 8 CPU cores and 16 GB of RAM. All our experiments simulate a single-cell scenario, since UEs compete for RAN resources only with users in the same cell. The CVSO may simultaneously manage more than one cell, and will have to solve the assignment problem for each cell independently.

We vary the number of *active* UEs simultaneously connected to a single cell and accessing the video service from 10 to 5000. (In practical 4G scenarios, the number of UEs in the connected state at the same eNodeB would not exceed a few hundreds.) For a given number of connected users, we report mean values over 200 executions of each algorithm with 95% confidence intervals. In each execution, the CQI value for each user is drawn uniformly at random from the $[1, 15]$ range. Our experiment configuration is summarized in Table 3. The selected genetic algorithm parameters were empirically shown to achieve a good trade-off between execution time and solution quality in our settings.

Table 3: Experiment configuration

General settings	
Channel bandwidth	20 Mhz (100 PRBs)[a]
Number of video representations	$\{4, 6\}$
Number of video clients	10-5000
Number of iterations	200
Confidence intervals	95%
Video characteristics	
Bitrates (Kbps)	117, 238, 487, 977, 1955, 3901
Corresponding MOS values	$1.07, 1.43, 2.69, 4.18$, 4.81, 4.96
GA parameters	
Solution pool size	40
Generations to convergence	15[b]
Maximum number of generations	30
Crossover rate	0.5
Mutation rate	0.0[c]

[a] In all our experiments, we assume that all 100 PRBs are available to the video streaming service.

[b] Number of consecutive generations for which the fitness value of the best candidate solution does not change significantly. After that, the GA is considered to have converged and terminates. Otherwise, the algorithm terminates when a maximum number of generations has been reached.

[c] After experimenting with various configurations, we found that mutation does not offer significant advantages and thus we do not apply it.

The branch-and-cut algorithm implemented using the ILOG CPLEX optimization suite[19] is an exact one, thus it derives the optimal solution. On the other hand, being heuristics, the genetic algorithm (GA) and the baseline one may fail to reach the optimal. Figure 6 presents the

[17] The source code of these algorithms and tools to reproduce our experiments can be found here: `https://github.com/pfrag/vassign`

[18] Assuming no interruptions, i.e., only considering the effects of the QP on picture quality.

[19] `https://www.ibm.com/products/ilog-cplex-optimization-studio`

absolute value of the objective function (sum of QoE) of the solutions returned by each algorithm. We note that for congested scenarios (e.g., when there are thousands of UEs), all three algorithms derive the optimal solution, which is however one where a small ratio (decreasing as the number of UEs grows) of UEs receive video, and this is of the lowest possible quality. Figure 7 compares the two heuristic algorithms in terms of how well they approach the optimal solution for a scenario where all 6 video representations with the characteristics presented in Table 3 are available. The metric of interest is the ratio of the objective function value of the solution returned by a heuristic compared with the optimal. It can be observed that it is in the realistic scenarios where the optimality gap is larger. It should be pointed out, though, that in all the cases we evaluated, the solutions returned by the GA have on average a value of 0.89 of the optimal or higher, while the baseline returned solutions as low as 0.5 of the optimal (for scenarios with 70-80 UEs). After a certain point, namely beyond 500 users, the performance of the two heuristic solutions becomes almost identical (but still suboptimal), while for more than 1300 users both the GA and the baseline practically always manage to return the optimal solution, and thus the optimality ratio reaches 1.0.

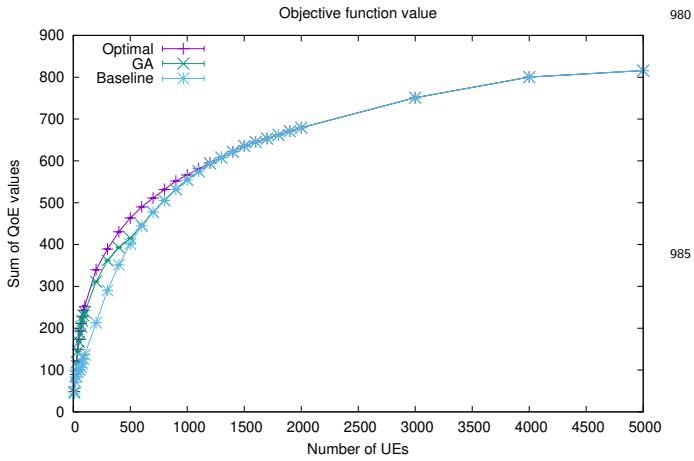


Figure 6: Comparison of the three algorithms in terms of the absolute objective function value. Six video representations are available.

The behaviour of the three algorithms is qualitatively

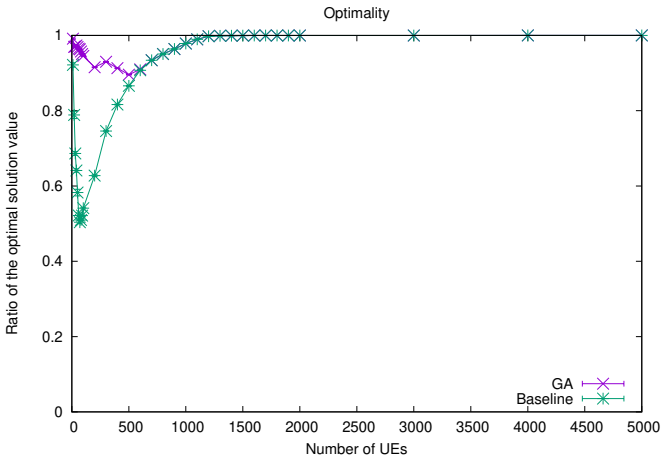


Figure 7: Solution quality of the GA and the baseline heuristics. The reported values are the ratio of the objective function value returned by each of the two algorithms to that of the optimal solution returned by the branch-and-cut algorithm of CPLEX. Six video representations are available.

different and this is demonstrated by the number of UEs that each algorithm eventually serves with (any) video. Due to its operation, i.e., starting by assigning to as many users as possible the minimum quality video and progressively attempting to improve their quality, the baseline algorithm tends to maximize the number of UEs that get at least *some* video. One the other hand, in many of the realistic cases (10-500 UEs) the GA leaves significantly more UEs without *any* video.[20] This is demonstrated in Figure 8. At the same time, having as an objective to maximize the sum of QoE values, the optimal solution strikes a balance between the two: If we also consider the average QoE across only the *served* UEs, i.e., the ones that are admitted to the video streaming service, shown in Figure 9, the optimal solution offers a significantly better QoE than

[20] To an extent, this is related with the way we generate the pool of initial GA solutions. In particular, the initial solution pool includes (i) randomly-generated solutions, (ii) solutions where as many UEs as possible as covered with low-quality video, and (iii) solutions where only the representation of the highest possible quality is assigned, and thus fewer UEs are served. In low-congestion scenarios the genes of the latter solutions have good chances to survive across generations, and in such cases the GA may promote solutions where few UEs are admitted, but with high MOS.

the baseline while serving a significantly higher number of UEs than the GA for the scenarios with up to 500 UEs.

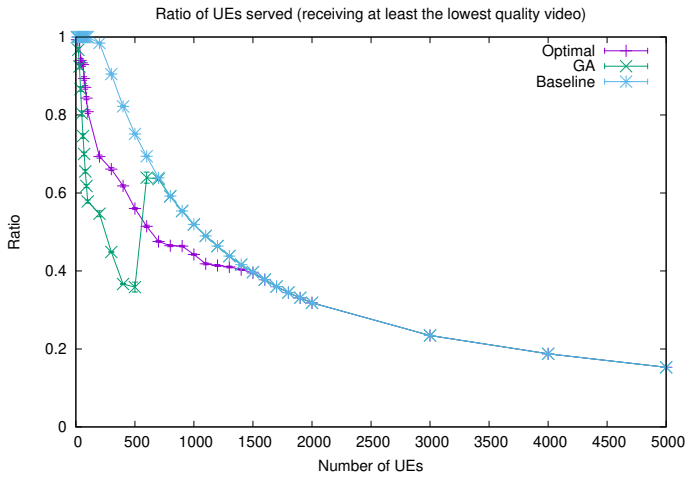
The reader may observe that in scenarios with large numbers of users, the quality of the video they enjoy is the minimum possible. In Figure 9a, for example, where all 6 representations are available, for large numbers of UEs the average MOS across all served users is 1.07, which is unacceptable for most infotainment services. It is up to the video service provider to control that. For instance, Figure 9b shows the average quality achieved across all served users when the service provider restricts the set of available representations to only those that are characterized by a MOS of 2.69 or more (i.e., the top four of the set of our available representations). Since only the highest-bitrate video representations are available, the ratio of UEs served drops, as shown in Figure 8b. This is a trade-off that the vertical needs to address by tuning the available video qualities according to the specifics of the infotainment service. Further adaptations to our model and algorithms to account for individual user preferences could also be possible, where each user defines the minimum video quality that they accept.

As expected, the exact algorithm comes with an increased computational cost compared with the heuristic ones. However, the results shown in Figure 10 demonstrate that solving the problem to optimality takes less than 200-300 ms for most practical cases, which is significantly less than the `minimumUpdatePeriod`. The genetic algorithm and our simple baseline perform mach faster, at the expense of suboptimal solutions. For large problem instances, when the running time of the branch-and-cut algorithm may become a concern, the CVSO operator may select to switch to one of the heuristics, since they achieve identical performance (see Figure 7) with a much lower execution time (on average 232 ms and 17 ms in a scenario with 2000 UEs for the GA and the baseline algorithm, respectively). Therefore, whichever algorithm the operator selects to apply, the *CVSO* is able to adapt the video streaming bitrate of each user to the network conditions in a timely manner.

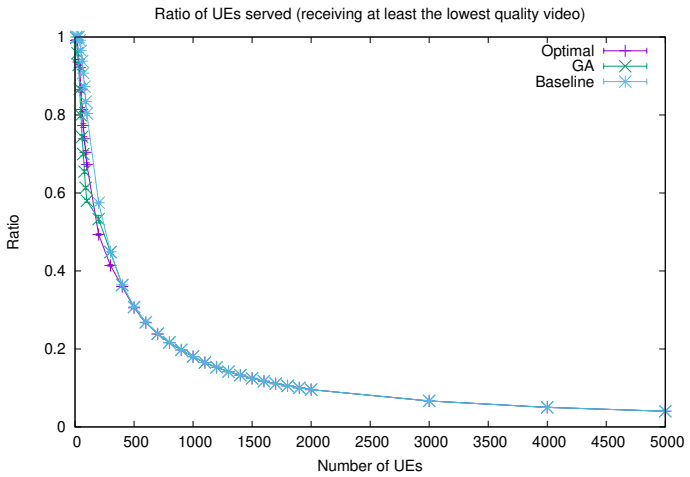
6. Discussion

There are some specific operational aspects that we have not addressed in this article. First, the objective function in our representation assignment model aims to maximize the sum of QoE values across the set of users accessing the video service. This, however, provides no *fairness* guarantees. For example, it is possible that it sacrifices on the viewing experience of some users, if serving others with higher quality contributes towards maximizing the global objective. It is a matter of the policy of the vertical service provider to select a different objective function which would offer better fairness performance. We defer fairness issues to future work.

Another issue that we do not address here is when and how often to execute the assignment algorithm. The optimal choice depends on the dynamics of the environment and the particular settings under which the system operates. In scenarios characterized by significant churn, as would be the case in a highly distributed environment where the many CVSO instances operate, each on a cell of relatively small size and with fast moving vehicles, the algorithm might have to be executed frequently, each time the VSS front end notifies the VSC that the set of users connected to a VSS instance changes. While this looks as a significant overhead, one has to bear in mind that with a small cell size comes a small set of connected users, on which our algorithms have been shown to execute fast. On the other hand, in an environment where few CVSO instances are responsible for large service areas, it may be acceptable to recalculate quality assignments periodically and at a lower frequency, even at multiples of the `minimumUpdatePeriod`. In this case, what matters is to reflect accurately the changes in channel quality, which is also a function of the mobility characteristics of the vehicles in the particular settings (e.g., urban scenarios vs.



(a) Six available video representations; minimum representation bitrate/MOS: 117Kbps/1.07.



(b) Four available video representations; minimum representation bitrate/MOS: 487 Kbps/2.69.

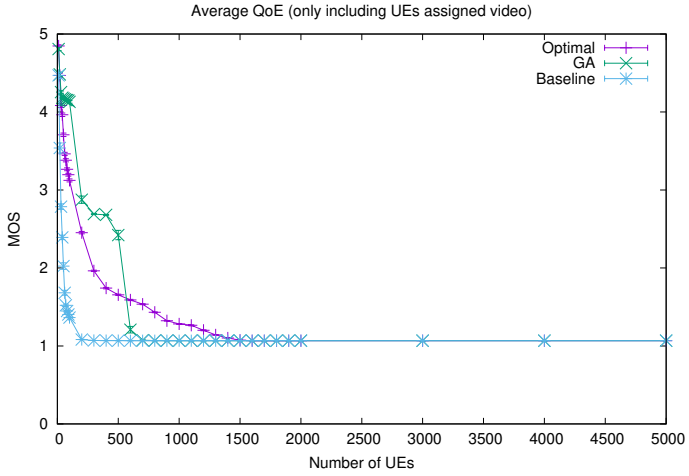
Figure 8: Comparison of the ratio of users which are admitted to the video streaming service, i.e., the ones that are assigned at least the minimum-quality video.

highways). Overall, selecting when to execute the algorithm, which algorithm to activate, and under which configuration, are design decisions that have to be taken by the vertical service provider given its operational environment. These are significant aspects that warrant further research.

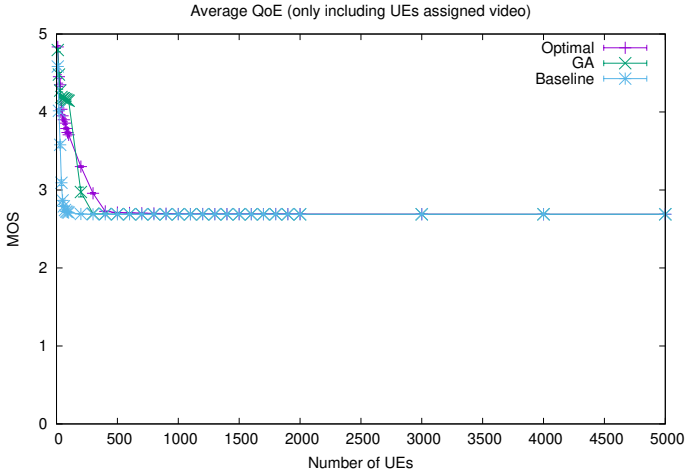
7. Conclusion

Taking advantage of the recent advances in the area of Multi-access Edge Computing, we presented the design and implementation of a Connected Vehicle Service Orchestrator that manages heterogeneous automotive applications at the mobile edge. Our system delivers QoE-optimized infotainment video services coexisting with services requiring a minimum capacity but high reliability (e.g., road safety), and does so in a fully standards-based way: Video is delivered using DASH technologies, which are widely supported, and the proposed server-side video quality optimizations take place in a way transparent to the clients. At the same time, the proposed CVSO is deployed on top of a standards-compliant MEC platform that we have implemented, which exposes a RNI API adhering to the recent ETSI specifications. Contrary to typical receiver-driven video bit rate adaptation mechanisms, it is our CVSO that decides on the optimal video quality per user, based on the latter's channel quality characteristics, which are available via the RNIS.

Via testbed experiments over a MEC-capable LTE network, we have shown our server-assisted video adaptation scheme to improve on user experience. Finally, we have proposed alternative algorithms to solve the video representation assignment problem which exhibit different trade-offs between optimality and execution time. By experimenting with problem instances that vary widely in size, we explored these trade-offs and presented results that can be used by the service provider to dynamically select the algorithm to activate to attain specific response time targets. The solution to the problem can be derived fast even for large numbers of users, which combined with our full system implementation and testbed experiments demonstrate the feasibility of our architecture and its suitability for MEC deployment.



(a) Six available video representations; minimum representation bitrate/MOS: 117Kbps/1.07.



(b) Four available video representations; minimum representation bitrate/MOS: 487 Kbps/2.69.

Figure 9: Comparison of the average QoE across all the users which are admitted to the video streaming service.

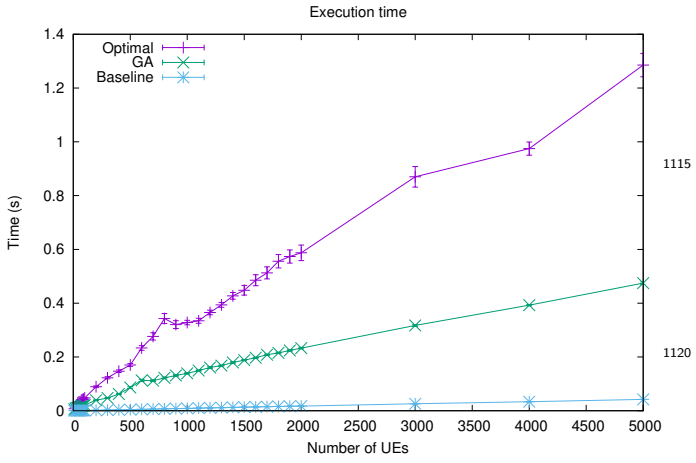


Figure 10: Running time comparison. Six video representations are available.

Acknowledgment

This work has been partially funded by the EC H2020 5G-Transformer Project (grant no. 761536).

References

[1] 5G automotive vision, White paper, 5GAA (Oct. 2015).

[2] S. Chen, J. Hu, Y. Shi, Y. Peng, J. Fang, R. Zhao, L. Zhao, Vehicle-to-Everything (v2x) Services Supported by LTE-Based Systems and 5G, IEEE Communications Standards Magazine 1 (2) (2017) 70–76.

[3] Toward fully connected vehicles: Edge computing for advanced automotive communications, White paper, 5GAA (Dec. 2017).

[4] ETSI GS MEC 003, Multi-access Edge Computing (MEC); Framework and Reference Architecture, v2.1.1 (Jan. 2019).

[5] ETSI GS MEC 012, Mobile Edge Computing (MEC); Radio Network Information API, v1.1.1 (Jul. 2017).

[6] I. Sodagar, The MPEG-DASH standard for multimedia streaming over the internet, IEEE MultiMedia 18 (4) (2011) 62–67.

[7] J. Fajardo, I. Taboada, F. Liberal, Improving content delivery efficiency through multi-layer mobile edge adaptation, IEEE Network 29 (6) (2015) 40–46.

[8] S. Salsano, L. Chiaraviglio, N. Blefari-Melazzi, C. Parada, F. Fontes, R. Mekuria, D. Griffioen, Toward superfluid deployment of virtual functions: Exploiting mobile edge computing for video streaming, in: Proc. SOFT5@ITC 2017, 2017.

[9] C. Ge, N. Wang, G. Foster, M. Wilson, Toward QoE-Assured 4K Video-on-Demand Delivery Through Mobile Edge Virtualization With Adaptive Prefetching, IEEE Trans. Multimedia 19 (10) (2017) 2222–2237.

[10] C. Ge, N. Wang, W. K. Chai, H. Hellwagner, QoE-Assured 4K HTTP Live Streaming via Transient Segment Holding at Mobile Edge, IEEE Journal on Selected Areas in Communications 36 (8) (2018) 1816–1830.

[11] Y. Li, P. A. Frangoudis, Y. Hadjadj-Aoul, P. Bertin, A mobile edge computing-assisted video delivery architecture for wireless heterogeneous networks, in: Proc. IEEE ISCC, 2017.

[12] D. Sabella, N. Nikaein, A. Huang, J. Xhembulla, G. Malnati, S. Scarpina, A hierarchical MEC architecture: Experimenting the RAVEN use-case, in: Proc. IEEE VTC Spring, 2018.

[13] Y. Tan, C. Han, M. Luo, X. Zhou, X. Zhang, Radio network-aware edge caching for video delivery in MEC-enabled cellular networks, in: Proc. IEEE WCNC Workshops, 2018.

[14] A. Mehrabi, M. Siekkinen, A. Ylä-Jääski, Edge computing assisted adaptive mobile video streaming, IEEE Trans. Mob. Comput. 18 (4) (2019) 787–800.

[15] S. Shi, V. Gupta, M. Hwang, R. Jana, Mobile VR on edge cloud: A latency-driven design, in: Proc. 10th ACM Multimedia Systems Conference (MMSys '19), 2019.

[16] G. Rigazzi, J. Kainulainen, C. Turyagyenda, A. Mourad, J. Ahn, An edge and fog computing platform for effective deployment of 360 video applications, in: Proc. IEEE WCNC Workshops, 2019.

[17] ETSI GS MEC 002, Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements, v2.1.1 (Oct. 2018).

[18] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration, IEEE Communications Surveys and Tutorials 19 (3) (2017) 1657–1681.

[19] C. I, Y. Liu, S. Han, S. Wang, G. Liu, On big data analytics for greener and softer RAN, IEEE Access 3 (2015) 3068–3075.

[20] J. Pérez-Romero, V. Riccobene, F. Schmidt, O. Sallent, E. Jimeno, J. Fernandez, A. Flizikowski, I. Giannoulakis, E. Kafetzakis, Monitoring and analytics for the optimisation of cloud enabled small cells, in: Proc. 23rd IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2018.

[21] G. Bianchi, E. Biton, N. Blefari-Melazzi, I. Borges, L. Chiaraviglio, P. de la Cruz Ramos, P. Eardley, F. Fontes, M. J. McGrath, L. Natarianni, D. Niculescu, C. Parada, M. Popovici, V. Riccobene, S. Salsano, B. Sayadi, J. Thomson, C. Tselios, G. Tsolis, Superfluidity: a flexible functional architecture for 5G networks, Trans. Emerging Telecommunications Technologies 27 (9) (2016) 1178–1186.

[22] T. Taleb, P. A. Frangoudis, I. Benkacem, A. Ksentini, CDN Slicing over a Multi-Domain Edge Cloud, IEEE Transactions on Mobile Computing (in press). doi:10.1109/TMC.2019.2921712

[23] ISO/IEC Standard 23009-5:2017, Dynamic adaptive streaming over HTTP (DASH) – Part 5: Server and network assisted DASH (SAND) (Feb. 2017).

[24] F. Giust, V. Sciancalepore, D. Sabella, M. C. Filippou, S. Mangiante, W. Featherstone, D. Munaretto, Multi-access edge computing: The driver behind the wheel of 5g-connected cars, IEEE Communications Standards Magazine 2 (3) (2018) 66–73.

[25] ETSI GR MEC 022, Multi-access Edge Computing(MEC); Study on MEC Support for V2X Use Cases, v2.1.1 (Sep. 2018).

[26] A. G. Olmos, F. V. Gallego, R. Sedar, V. Samoladas, F. Mira, J. Alonso-Zarate, An automotive cooperative collision avoidance service based on mobile edge computing, in: Proc. 18th International Conference on Ad-Hoc Networks and Wireless (ADHOC-NOW), 2019.

[27] G. Avino, P. Bande, P. A. Frangoudis, C. Vitale, C. Casetti, C. F. Chiasserini, K. Gebru, A. Ksentini, G. Zennaro, A MEC-based extended virtual sensing for automotive services, IEEE Transactions on Network and Service Management 16 (4) (2019) 1450–1463.

[28] ETSI EN 302 637-3, Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service, v1.3.1 (Apr. 2019).

[29] A. Keivani, F. Ghayoor, J.-R. Tapamo, Collaborative Mobile Edge Computing in eV2X: A Solution for Low-Cost Driver Assistance Systems, Wireless Personal Communications (2019).

[30] S. Maheshwari, W. Zhang, I. Seskar, Y. Zhang, D. Raychaudhuri, EdgeDrive: Supporting advanced driver assistance systems using mobile edge clouds networks, in: Proc. IEEE INFOCOM Workshops, 2019.

[31] Q.-H. Nguyen, M. Morold, K. David, F. Dressler, Car-to-Pedestrian communication with MEC-support for adaptive safety of Vulnerable Road Users, Computer Communications (2020) 83–93.

[32] A. Napolitano, G. Cecchetti, F. Giannone, A. L. Ruscelli, F. Civerchia, K. Kondepu, L. Valcarenghi, P. Castoldi, Implementation of a MEC-based Vulnerable Road User Warning System, in: Proc. 2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE), 2019.

[33] C. Campolo, A. Iera, A. Molinaro, G. Ruggeri, MEC Support for 5G-V2X Use Cases through Docker Containers, in: Proc. IEEE WCNC, 2019.

[34] C. Campolo, R. dos Reis Fontes, A. Molinaro, C. E. Rothenberg, A. Iera, Slicing on the road: Enabling the automotive vertical through 5G network softwarization, Sensors 18 (12) (2018) 4435.

[35] I. Afolabi, T. Taleb, P. A. Frangoudis, M. Bagaa, A. Ksentini, Network Slicing-Based Customization of 5G Mobile Services, IEEE Network 33 (5) (2019) 134–141.

[36] ETSI EN 302 637-3, Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service, v1.2.2 (Nov. 2014).

[37] J. Santa, F. Pereñíguez, A. Moragón, A. F. Skarmeta, Experimental evaluation of CAM and DENM messaging services in vehicular communications, Transportation Research Part C: Emerging Technologies 46 (2014) 98–120.

[38] ETSI TS 101 539-3, Intelligent Transport Systems (ITS); V2X

Applications; Part 1: Road Hazard Signalling (RHS) application requirements specification, v1.1.1 (Nov. 2013).

[39] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, K. P. Kontovasilis, Flexran: A flexible and programmable platform for software-defined radio access networks, in: Proc. ACM CoNEXT, 2016.

[40] S. Arora, P. A. Frangoudis, A. Ksentini, Exposing radio network information in a MEC-in-NFV environment: the RNISaaS concept, in: Proc. 5th IEEE Conference on Network Softwarization (NetSoft 2019), 2019.

[41] ISO/IEC Standard 23 009.1:2014, Information Technology - Dynamic Adaptive Streaming over HTTP (DASH) - Part 1: Media Presentation Description and Segment Formats (May 2014).

[42] E. Thomas, M. O. van Deventer, T. Stockhammer, A. C. Begen, M. L. Champel, O. Oyman, Applications and deployments of server and network assisted DASH (SAND), in: Proc IBC, 2016.

[43] ETSI GS MEC 010-2, Mobile Edge Computing (MEC); Mobile Edge Management; Part 2: Application lifecycle, rules and requirements management, v1.1.1 (2017).

[44] ETSI GS MEC 016, Multi-access Edge Computing (MEC); Device application interface, v2.2.1 (2020).

[45] D. Sabella, V. Sukhomlinov, L. Trang, S. Kekki, P. Paglierani, R. Rossbach, X. Li, Y. Fang, D. Druta, F. Giust, L. Cominardi, W. Featherstone, B. Pike, S. Hadad, Developing software for Multi-access Edge Computing, White Paper 20, ETSI (Feb. 2019).

[46] ETSI GS NFV-MAN 001, Network Functions Virtualisation (NFV); Management and Orchestration, v1.1.1 (Dec. 2014).

[47] ETSI GS MEC 011, Mobile Edge Computing (MEC); Mobile Edge Platform Application Enablement, v1.1.1 (Jul. 2017).

[48] 3GPP, Technical Specification (TS), 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures, Version 14.2.0 (Nov. 2014).

[49] ITU-T Recommendation, P.911, Subjective Audiovisual Quality Assessment Method for Multimedia Applications (Dec. 1998).

[50] H. Kellerer, U. Pferschy, D. Pisinger, Knapsack problems, Springer, 2004. doi:10.1007/978-3-540-24777-7.

[51] J. E. Mitchell, Branch-and-Cut Algorithms for Combinatorial Optimization Problems, in: Handbook of Applied Optimization, Oxford University Press, 2002, pp. 65–77.

[52] M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, Cambridge, MA, USA, 1998.

[53] P. Schmitt, B. Landais, F. Y. Yang, Control and User Plane Separation of EPC nodes (CUPS), Tech. rep., 3GPP (Jul. 2018). URL http://www.3gpp.org/cups

[54] S. Lederer, Optimal Adaptive Streaming Formats MPEG-DASH & HLS Segment Length, https://bitmovin.com/mpeg-dash-hls-segment-length/.

[55] J. Allnatt, Transmitted-Picture Assessment, Wiley, 1983.

[56] ITU-T Recommendation, P.1201 Standard, Parametric non-intrusive assessment of audiovisual media streaming quality. Amendment 2: New Appendix III - Use of ITU-T P.1201 for non-adaptive, progressive download type media streaming (Dec. 2013).

[57] M. N. Garcia, R. Schleicher, A. Raake, Impairment-Factor-Based Audiovisual Quality Model for IPTV: Influence of Video Resolution, Degradation Type, and Content Type, EURASIP Journal on Image and Video Processing 2011 (1). doi:10.1155/2011/629284.

[58] ITU-T Recommendation, G.107 Standard, The E-model: A computational model for use in transmission planning (2013).

[59] G. Rubino, Quantifying the Quality of Audio and Video Transmissions over the Internet: The PSQA Approach, in: J. A. Barria (Ed.), Communication Networks & Computer Systems, Imperial College Press, 2005.

[60] K. D. Singh, Y. Hadjadj-Aoul, G. Rubino, Quality of Experience estimation for Adaptive HTTP/TCP video streaming using H.264/AVC, in: Proc. IEEE CCNC, 2012.

[61] T. Groléat, M. Sbai, S. Vaton, Y. Hadjadj-Aoul, K. Singh, S. Moteau, Advances on monitoring primitives and integration in the VIPEER prototype, ANR VIPEER project deliverable D2.3 (Dec. 2012).