TOWARDS FAIRNESS AND QOE-BASED EDGE ALLOCATION FOR MULTIPLAYER VIRTUAL REALITY APPLICATIONS IN EDGE COMPUTING

Athanasios Tsipis[1], Vasileios Komianos[2], Konstantinos Oikonomou[3], Ioannis Stavrakakis[4]
[1]Dept. of Digital Media & Communication, Ionian University, Kefalonia, Greece, atsipis@ionio.gr, [2]Dept. of Audio & Visual Arts, Ionian University, Corfu, Greece, vkomianos@ionio.gr, [3]Dept. of Informatics, Ionian University, Corfu, Greece, okon@ionio.gr, [4]Dept. of Informatics & Telecommunications, National and Kapodistrian University of Athens, Athens, Greece, ioannis@di.uoa.gr

NOTE: Corresponding author: Athanasios Tsipis, atsipis@ionio.gr

Abstract – *Edge computing has emerged as the next big thing in distributed computing, by extending the cloud paradigm and offering efficient ways to engage with latency-intolerant applications, such as Virtual Reality (VR) multiplayer games. In edge computing, the service providers can benefit from existing cellular infrastructure to deploy services on edge servers that reside in close proximity to the users. Given the limited available budget for edge resource investment, one fundamental problem that manifests is the discovery of a prudent edge allocation strategy, that will efficiently prescribe which users are assigned to which edge servers, in order to tackle application-specific requirements, like minimizing system deployment costs. In this paper, considering the frequent interactions and view inconsistencies occurring among multiple users immersed in the same VR game, we address the problem from the users' perspective, focusing on improving their edge admission rate, resource provisioning and overall fairness, in order to subsequently maximize the average Quality of Experience (QoE). We call this the "Fairness and QoE-Based Edge Allocation" (FQEA) problem, formally formulating its properties and theoretically proving its complexity. However, discovering optimal solutions to the NP-hard FQEA in large-scale VR scenarios is challenging. Hence, we propose FQEA-H, a heuristic algorithm to generate allocation strategies in reasonable time. Comprehensive simulations, conducted on a real-world topological trace, demonstrate how FQEA-H can tackle the problem effectively, generally outperforming both the baseline and state-of-the-art alternatives.*

Keywords – Edge computing, fairness, quality of experience, user edge allocation, virtual reality

1. INTRODUCTION

From the onset of the COVID-19 pandemic, the global market of multiplayer games and Virtual Reality (VR) has experienced a massive surge, standing out among other industries, which in many cases had no choice but to scale down their activity as a result of worldwide lockdowns [1]. While this has generated great hype towards immersive technologies, leading to estimates that project VR game service provisioning profits to reach up to 45 billion USD by 2025 [2], the VR adoption rate is still relatively low, and user experience is recognized by many as one of the major barriers to its wide proliferation.

Quality of Experience (QoE) is impacted by a number of factors that range from application to user-specific. For VR, it is undeniable that high end-to-end *interactivity* is a key enabler for achieving the desired levels of QoE [3]. The need becomes even more profound when it comes to applications envisaged for the realization of the new metaverse vision (e.g., open-world social VR and massively multiplayer cloud games), wherein a plethora of users are expected to heavily cooperate, continuously exchange data, and jointly participate in collaborative virtual events on a frequent basis [4].

However, despite existing online games being able to support a relatively high number of users, albeit with high specification system requirements, for future-generation multiplayer VR games, achieving high interactivity is still a nascent concept even for the mature cloud game vendors [5]. This is mainly attributed to the prohibitive bandwidth for streaming VR frames of high graphics quality (e.g., panoramic, 4k, or 360° videos) and the stringent delay requirements for responding to players' control actions in a timely fashion. Both of these aspects essentially necessitate the push of VR computation (specifically the game scene update and content rendering) away from remote clouds and close to the actual source of data creation, i.e., near the end users [6].

Thankfully, this notion seamlessly overlaps with the emerging paradigm of Mobile Edge Computing (MEC) [7], wherein Service Providers (SPs) extend their service provisioning capabilities with intelligent edge functionality that resides near the users' immediate neighborhoods. By caching content and running computation on existing cellular infrastructure rented from telecommunication carriers, e.g., Base Stations (BSs), SPs can effectively mitigate the dangers of remote video sequence streaming [8]. Users, within the coverage range of the BSs, can then interact with the deployed edge servers and gain access to the offered VR game applications (usually via thin clients) [9].

Unfortunately, this paradigm shift is non-trivial. On the one hand, it may pave the way for new methods of user engagement but, on the other, it also opens the door to new and unforeseen challenges. One fundamental problem, given the SPs' finite *budget* in edge infrastructure investment, is the discovery of an intelligent user-server allocation strategy that can optimally assign as many users as possible to a limited number of available edge servers. However, for immersive social VR games, where the virtual world is shared between multiple users, conventional edge allocation models are deemed increasingly inefficient. This is because edge allocation, solely on the basis of independent users, only considers user-server delays, rather than the collective experience, and, thus, remains oblivious to inter-player *interactivity* [10]. The latter is critical in nourishing *fairness* for multiplayer VR, in order to achieve view consistency among game scenes observed separately, but simultaneously, by various individuals [11].

Generally speaking, inter-player latency is measured by the time it takes to traverse the interaction delay path connecting any two users. As such, it consists of three main parts [12]. Initially, a user u issues an in-game oper- ation, which is sent to its associated edge server s. There, after necessary computation (i.e., processing/ rendering), s updates the VR scene that captures the new game progress. The resultant frame is in sequel streamed back to u for display, but also forwarded to other edge servers $s' \neq s$, before finally being delivered to their own as- signed users $u' \neq u$. As this process involves multi- ple users over multiple edge servers, where the different users must witness the same scene changes in parallel, any judicious allocation strategy should in fact consider all entities involved in the interaction sequence when ad- dressing interactivity.

In essence, this suggests that users engaged with the same social VR game should connect to the same edge servers, or at least be assigned to edge servers that minimize their inter-player interaction latency, in order to gain their long-term engagement loyalty and enhance their immersion quality. Notwithstanding the criticality of the issue in player churn [13], discovering optimal solutions is not straightforward when considering the coverage range of the BSs in combination with the bounded computing capacity of the servers. The latter, in edge computing (contrary to its cloud counterpart), is characterized by limited available resources that need to be elastically provisioned and shared between the users, in order to maximize their Quality of Service (QoS), and in turn their perceived QoE [14].

Different from past research, where the focal point of interest was on the SPs' side, in order to tackle application-specific requirements like system cost minimization (e.g., [15]), in this paper, we address all previous challenges purely from the users' standpoint. First, we aim at maximizing the number of users admitted to the MEC network, since the assignment to remote clouds by default significantly degrades their immersion in terms of interactivity. Second, we aim at minimizing the fairness loss induced by view inconsistencies in the VR in-game scenes. To this end, we propose a clear methodology to reduce the minimum time required for users playing the same game to display the newly rendered frames. Third, we aim at optimizing QoS, given the servers' constraints and resource provisioning. By combining all aforementioned aspects, our central objective becomes the maximization of the users' average perceived VR QoE. Based on our analysis, we then seek an *allocation strategy* for the efficient mapping of users to edge servers. Obviously, this stipulates a complex optimization problem, which is hereinafter referred to as the *"Fairness and QoE-Based Edge Allocation"* (FQEA) problem.

Our contributions are threefold. In brief:

- We analytically model the properties of the MEC system in terms of edge provisioning and VR group gaming, and formally formulate the FQEA problem. Our goal, as stated, is to maximize the users' QoE on the basis of their edge admission rate, server resource provisioning, and game fairness. To the best of our knowledge, this is the first work to consider simultaneously all these parameters under multiple VR games hosted on the edge. Further, we theoretically prove that FQEA corresponds to an NP-hard problem, which makes its solution especially challenging.
- To efficiently address the high complexity in large-scale deployment scenarios, we propose FQEA-H, a *heuristic* algorithm to find allocation solutions in tractable time. FQEA-H is comprised of two independent phases, the first being the strategic assignment of the users, and the second entailing their QoS refinement in order to improve the overall QoE. Additionally, we analyze FQEA-H's convergence capability and prove its time complexity.
- Finally, we conduct extensive simulations, using a real-world topological trace, wherein we showcase the superiority of our algorithm in finding trade-offs among FQEA's targets, consistently outperforming both the state-of-the-art and baseline alternatives in maximizing QoE.

The remainder of the paper is organized as follows: Section 2 includes related work; Section 3 models the MEC system; Section 4 formulates the FQEA; Section 5 presents our algorithm; Section 6 includes numerical results that evaluate the performance of FQEA-H; and, lastly, Section 7 concludes our findings.

2. BACKGROUND WORK

With the increasing momentum of (mobile-) edge computing, significant research activity has been observed on many aspects relating to the efficient user edge allocation and interactivity enhancement [7].

2.1 Edge user-server allocation

From the perspective of the SP, the efficient handling of the available edge infrastructure is critical in offering elastic server/service provisioning. The research in [16] dealt with the problem of user edge assignment under capacity constraints, wherein the primary goal was to maximize the number of allocated users to the edge, while the secondary goal was to minimize the number of required edge servers in order to maximize the total profit. To address the two objectives, the authors utilized a lexicographic goal programming method and proposed a heuristic algorithm targeted at the strategic management of the users' resource demands. Recently, in [17], the aim shifted to the online decision version of the edge user allocation problem under mobility considerations, where users arrive and depart dynamically. The authors devised a decentralized reactive approach, to simultaneously address the users' admission rate, and the servers' hiring cost and total energy consumption. These studies find relevance to this work, in that both of them target (among others) the maximization of the users' MEC admission. Despite this being done by considering the rental cost inflicted by the servers' utilization, neither of them takes into account any budget limitations that SPs may have.

Proximity to edge servers is another crucial parameter when considering application availability in the MEC. It practically relates to the coverage range of the BSs hosting the edge servers. The subject was investigated in [18], where the authors combined user and SP expectations to present an approximation approach, that benefits from the overlapping coverage zones of different BSs, to balance the trade-off between the users' connection acceptance and the overall network robustness, in terms of accountability to edge server failures and impairments. Recently, in [19] the authors presented a nature-inspired optimization algorithm that acknowledges both proximity and capacity constraints to maximize user allocation while employing minimum edge servers. Likewise, in this research, proximity constraints are also considered and included as part of the final allocation solution to reduce unfairness caused by user-server interaction latency, by acknowledging BSs of a diversified coverage range.

With respect to resource provisioning, the edge allocation problem deals with the partition of the edge infrastructure to determine the specific amount of edge computing resources allocated for each user. The study in [20] presented a cost-effective resource allocation framework for the cloud-assisted edge computing environment, dynamically optimizing the computation capacity of edge nodes. Likewise, the recent research in [21] also attempted to minimize the overall cost for interactive applications running on the fog-assisted edge. The cost was impacted by the users' latency and servers' overall deployment needs. To address the issue, the authors designed a distributed algorithm to manage the users' generated load in terms of resource demands, and then placed edge rendering services in key BS locations within the network, respecting the delay intolerance and network congestion of VR and gaming applications. Similar to the approach adopted here, these solution benefit from the prudent management of the edge infrastructure and trade-off between the users' demands and the available edge resources before pushing service requests towards the remote cloud, which is unreliable in terms of interactivity and availability. Though the current research also employs the remote cloud for users not hosted at edge servers, the vantage point lies in that the former attempt to optimize QoS parameters relevant mainly to the SPs (e.g., tenancy cost), while the latter deals with the issue principally through the perspective of the users with the outlook of increasing their loyalty and engagement.

Concentrating on VR services, the work in [15], modeled the issue of social VR placement in edge computing as a combinatorial optimization problem for the minimization of several costs, including activation, proximity and colocation of services on edge servers. Based on their analysis, the authors proposed a graph-cut solution that makes expansion moves to alter the assignment and reduce the overall cost for the SP. Towards QoE maximization, the work in [22] correlated QoS and QoE metrics, and then utilized a greedy-like approach to administrate the edge servers' resources and solve a QoE-driven edge allocation problem via user reallocation. The same problem was extended in [23], jointly considering local computation on Internet of Thing (IoT) devices and computational offloading onto MEC servers. The authors therein adopted a game-theoretic approach to balance the trade-off and optimize QoE. In contrast to the above solutions, in this article, a key contributing element resides with the fact that QoE is affected by both social aspects among the users and individual features correlating to their QoS.

2.2 Interaction-driven edge allocation

Fueled by the need for increased interactivity and fairness in future-generation social interactive applications, the research focus is gradually shifting beyond inter-server latency, towards optimizing inter-player delays, when assigning users to servers. Initial attempts on this aspect, involve the work in [24] and [25], wherein the authors analyze the server provisioning problem as a means of achieving high interactivity and game-state synchronization. A significant finding in their research is that nearest server assignment strategies are far from optimal when it comes to total interaction time reduction among the users. To optimize fairness, the authors in [26] proposed a dynamic approach, that greedily computes for each newly-joint user the maximum interaction path length with all previously allocated users, and ultimately allocates the new user to the server that produces the minimum one. These studies motivate the current model, but do not consider the traits of the MEC ecosystem that naturally limit the server provision and user connectivity.

On the other hand, on the edge computing front, a characteristic example of inter-player optimization is traced back to [27], where the authors considered the Field of View (FoV), bandwidth availability and frame rendering demands of the users to propose an edge computing allocation solution that reduces the absolute delay difference in multiplayer VR. In comparison, in [28] the authors utilized reinforcement learning to induce fairness among players participating in the same game session. By making intelligent user-server matching, the authors minimized interaction latency variance in edge computing environments, while taking into account the capacity limitation of the involved servers. Both mentioned approaches embed upper latency thresholds in their optimization model regarding each player's absolute delay, yet the first considers the presence of only one MEC server and the second does not account for the finite budget of the SPs nor the QoE of the users.

Understanding social relations is critical in capturing interaction dependencies among users, as revealed in [12], where the authors designed an efficient algorithm for edge service placement, under a given upper deployment budget threshold. Their aim was to achieve low-delay pairwise interactions between the users, who were attributed different interaction frequencies based on prior observations. In [29], the authors proposed an edge assignment method that manages both network conditions and resource allocation, in order to balance the interaction delay among users belonging to the same cloud gaming group. On a similar note, the authors in [30] adopted a proactive deep learning assignment approach, in SDN-assisted edge systems, to estimate the amount of traffic originating from a specific user group, based on historical data. Subsequently, they employed a routing path allocation algorithm to improve fairness among the users, capitalizing on the knowledge acquired during the first stage. The work in [2] dealt with the dynamic service placement for VR group gaming. The authors formulated the problem, considering a combination of delay costs that impact, on the one hand, the pairwise communication between interactive users and their intermediate edge renderers and, on the other hand, the latency for in-group synchronization. Finally, the authors in [31] and later in [32], devised a heuristic algorithm for jointly optimizing both social interactivity and server provisioning for interactive multiplayer games hosted in the MEC. Their analysis showcased the necessity of capturing social aspects to augment the interaction among participants sharing common gaming trademarks. Despite the fact that these studies clearly point out that interactivity is indeed affected by social interactions transpiring over the longest interaction path length among the interactive participants, when it comes to game fairness, as it is shown in Section 4, we should also consider the shortest one, for this transforms the allocation problem into a multivariate one in terms of inter-player latency and gaming lag variability.

Common ground in all aforementioned studies is the necessity of discovering edge assignment strategies that strategically allocate resources to yield either high fairness or enhanced QoS/QoE, while abiding by proximity, capacity or cost-budget constraints. Thus, our work also falls under the design of fair and QoE-driven user-edge server matching algorithms. However, different from these approaches that dealt with interactivity focusing on either fairness or QoS as independent factors for achieving the necessary levels of QoE, the novelty behind this article rests with strictly binding fairness to QoS, and then encapsulating both of them within the FQEA optimization model itself. To do so, by acknowledging all prescribed constraints, we address the issue from the angle of the users, who are attributed diversified personal expectations in regards to their perceived VR graphics quality as well as high group expectations in terms of their in-game view consistency.

3. SYSTEM MODEL

In this section, we describe the fundamental attributes of our considered system model. The notations used throughout this paper are summarized in Table 1.

3.1 Preliminaries

Typically, in MEC the SP can leverage infrastructure of existing Metropolitan Area Networks (MANs) to deploy the required edge functionality, by renting the BSs from appropriate telecommunication carriers [12]. In this way, the MEC system essentially forms an overlay over the MAN itself, which can then be represented by an undirected graph $G = (V, E)$. Let $S = \{s_1, s_2, \dots, s_J\} \subseteq V$ denote the set of available edge servers that are colocated with the MAN's BSs. The edge servers can benefit from the coverage range of their attached BSs to provide covered users with VR applications. With a slight abuse of notation, we use s_j $(1 \leq j \leq J)$ to also signify the potential location for the deployment site of a VR gaming application instance. We hypothesize that there exists a set $H = \{h_1, h_2, \dots, h_K\}$, where $K = |H|$, of different candidate VR games to be hosted on various edge servers. Given these VR applications, let $U = \{u_1, u_2, \dots, u_I\} \subseteq V$ define the set of incoming users that wish to engage with their preferred h_k $(1 \leq k \leq K)$, by accessing the nearby edge servers, where $I = |U|$. Finally, let $E \subseteq V \times V$ denote the set of edge links that are established among the users and edge servers.

Different from conventional single-player games where the players act alone, users in multiplayer VR games can form clans and frequently exchange data, communicate through instant messaging or participate in collaborative virtual events. For instance, they can go on com-

Table 1 – Summary of main notations in order of appearance.

Parameter	Definition	Parameter	Definition
$G = (V, E)$	MEC network graph	$\mathcal{L}^{\mathcal{A}}(p, q)$	Interaction latency between nodes p and q under $\mathcal{A}$
S	Set of edge servers	$\mathcal{R}$	Remote cloud infrastructure
H	Set of VR games	$\pi^k_{i,n}$	Lag difference between u_i and u_n for h_k
U	Set of users	$t^k_n(i)$	Wait-time period of u_n after operation initiated by u_i in h_k
s_j	The jth server	$\hat{\tau}^{\mathcal{A}}_i, \check{\tau}^{\mathcal{A}}_i$	Maximum and minimum interaction latency of u_i under $\mathcal{A}$
h_k	The kth VR game	$\mathcal{V}^{\mathcal{A}}_i(k)$,	View inconsistency of u_i when playing h_k under $\mathcal{A}$
u_i	The ith user	$\varepsilon^{\mathcal{A}}_i$	QoE level of u_i under $\mathcal{A}$
U^k	Subset of users immersed in h_k	Q	Maximum QoE level
$\mathcal{A}$	Edge allocation strategy	$q^{\mathcal{A}}_i$	QoS level of u_i under $\mathcal{A}$
$s^{\mathcal{A}}_{j,i}$	The s_j that serves u_i under $\mathcal{A}$	$l^{\mathcal{A}}_i$	QoV level of u_i under $\mathcal{A}$
ϱ_j	Coverage range of the BS attached to s_j	ω_1, ω_2	Parameters for tuning the QoS-QoE correlation
$\lambda_{i,j}$	Distance between u_i and s_j	z^k_i	VR game user engagement indicator
r_i	Resource demands of u_i	S^o	Subset of open edge servers
C_j	Computing capacity of s_j	$S^o_{u_i}$	Subset of u_i's neighbor edge servers
r_i	Resource demands of u_i	$S^o_{u_i}$	Subset of u_i's neighbor open edge servers
$\alpha^{\mathcal{A}}_{i,j}$	User-server assignment indicator	c_j	Residual capacity of server s_j
B	SP's available budget	$U^{\mathcal{A}}_S$	User edge admission rate under $\mathcal{A}$
$\beta^{\mathcal{A}}_j$	Server utilization indicator	$\mathcal{F}^{\mathcal{A}}$	Fairness loss under $\mathcal{A}$
$d(p, q)$	Network delay between nodes p and q	$\mathcal{E}^{\mathcal{A}}$	Users average perceived QoE under $\mathcal{A}$

mon quests and fight together hostile bosses, or compete against each other in specially configured battle arenas. In essence, these users are characterized by recurrent interactions and thus constitute a *game group*. Let $U^k \subseteq U$ define the subset of all users engaged with the same game $h_k \in H$. For convention, in this paper, each game group refers to the users immersed in a particular VR game hosted on the MEC servers. Still, this assumption can be trivially expanded to further separate users who share the same game into additional classes on the basis of their in-game preferences and interactions. For instance, users meeting in different VR meeting rooms, using however the same VR application, users that enter the VR game world under similar configurations (e.g., specific language settings), or users that belong to the same virtual clan and follow the same in-game storyline, can also be regarded as members with common attributes, and thus can be reorganized accordingly into different sub-groups within the application itself. Such divisions will entail no changes to the system model or the behavior of the FQEA and FQEA-H solutions presented in later parts of the paper. As such, for reasons of notational clarity, for the rest of the paper, we will only consider groups formed by users immersed in discrete games h_k ($1 \leq k \leq K$).

With that said, each user u_i ($1 \leq i \leq I$) then needs to be assigned to a specific edge server in order to send its operations and receive VR frame updates. Let $s^{\mathcal{A}}_{j,i}$ denote the edge server s_j serving user u_i under a determined allocation strategy, say $\mathcal{A}$. Next, we present the minimum criteria for any $\mathcal{A}$ to be considered as *valid*.

3.2 Proximity model

Modern MANs are characterized by substantial augmentations in wired communication, a fact that is even further highlighted under the 5G (and beyond) era. Inspired by relevant research, e.g., [33, 32], we assume that the edge servers are able to communicate with each other, through their BSs, over dedicated backhaul inter-server wired connectivity. Contrary to this, mobile users may access the deployed edge servers over wireless links that depend on the coverage range of their BSs.

Let ϱ_j denote the coverage range of the BS attached to edge server $s_j \in S$. Then, s_j is qualified to directly serve some user $u_i \in U$, i.e., create an edge link $e(u_i, s_j) \in E$, when their geographical distance is no more than ϱ_j. In other words, the following *proximity constraint* must be satisfied.

Definition 1 (proximity constraint). *$u_i \in U$ can access a VR game hosted by some $s_j \in S$, if and only if the following relation holds:*

$$\exists e(u_i, s_j) \in E : \lambda_{i,j} = ||u_i - s_j|| \leq \varrho_j, \quad u_i \in U, s_j \in S, \tag{1}$$

where $\lambda_{i,j} = ||u_i - s_j||$ defines their in-between distance given by some metric.

For the sake of simplicity, in this paper (like many previous in literature e.g., [28]), the $|| \cdot ||$ represents the Euclidean distance between u_i and s_j, although other (perhaps more sophisticated) metrics can also be employed without violating the general model or the execution of our proposed algorithm. If *proximity constraint (1)* is not satisfied for any s_j, then u_i ineluctably connects to the SP's remote cloud infrastructure, denoted here as $\mathcal{R}$, which serves as the default VR game server for users that are unable to connect to any nearby edge servers.

3.3 Edge provisioning model

In MEC, the physical resources of the edge servers, in terms of computing/rendering (e.g., GPU, CPU, memory, etc.), are shared between their assigned users. However, different from conventional cloud-centric computing models, where the resources of the data centers are assumed to be practically infinite, the computing capacity of the edge servers is constrained by finite sizes [14]. Let C_j denote the capacity limit in regards to resource availability of $s_j \in S$, beyond which its assigned users begin to experience losses in frame quality and increased interaction delay that can cause VR vertigo.

Aside from the servers, users also, depending on their devices' display configuration capabilities and their own gaming preferences, usually have different resource requirements in terms of Quality of Video (QoV) [21]. Generally speaking, QoV is hard to assess due to subjective factors pertaining to the users' individual gameplay expectations. Still, one of the major elements reported in literature to impact QoV is the actual graphics quality, which can be application-dependent [34, 35]. For instance, a user playing a fast-paced shooter VR game, will probably be positively inclined towards lesser QoV that yields lower interaction delays, whereas a user engaged with a slow-paced role-playing VR game, most likely spends more time exploring the game scenery or enjoying the landscapes rather than participating in serious battles, and thus is regarded as more delay tolerant and should be offered higher QoV. In parallel, a u_{HMD}, who is immersed in the virtual world via a specialized VR Head Mounted Display (HMD), can support high graphics quality but requires higher frame rates at the expense of an increased number of edge resources. In contrast, a u_{ST}, who is engaged with the VR game via a smart tablet (ST), will most likely be satisfied with smaller image resolutions that demand lower rendering power and, thereby, lesser edge resources. To quantify these types of relationships, we use r_i to denote the amount of absolute resource demands required by some u_i to gain the desired QoV.

To guarantee the prudent resource provisioning of the system and avoid over-utilization, the following *capacity constraint* must be satisfied by all edge servers.

Definition 2 (capacity constraint). *$s_j \in S$ can efficiently support, in terms of computing and rendering performance, its assigned users, that meet proximity constraint (1), if, and only if, the following relation holds:*

$$\sum_{i=1}^{I} r_i \alpha_{i,j}^{\mathcal{A}} \leq C_j, \quad j \in \{1, \dots, J\}, \tag{2}$$

where $\alpha_{i,j}^{\mathcal{A}} \in \{0, 1\}$ is a binary index showing whether u_i is assigned to server s_j under $^{\mathcal{A}}$; that is, $\alpha_{i,j}^{\mathcal{A}} = 1$ if the condition is true, while $\alpha_{i,j}^{\mathcal{A}} = 0$ otherwise.

Still, even if the capacity of the edge servers is not infringed, in realistic deployment scenarios the SP will most

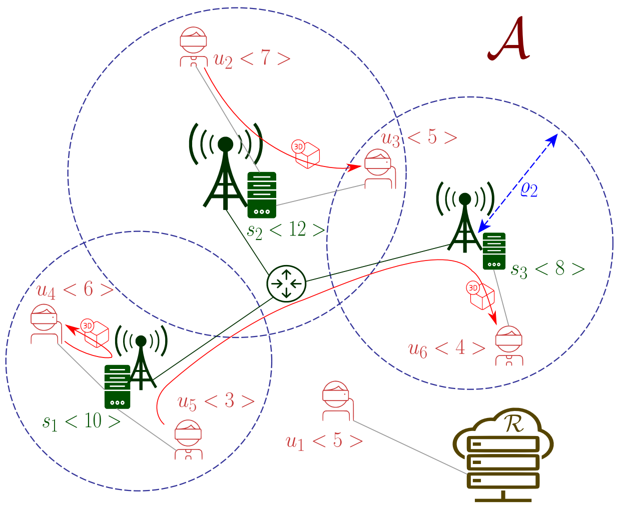


Fig. 1 – An overview of the edge interaction processes under allocation $\mathcal{A} = \{s_{\mathcal{R},1}^{\mathcal{A}}, s_{2,2}^{\mathcal{A}}, s_{2,3}^{\mathcal{A}}, s_{1,4}^{\mathcal{A}}, s_{1,5}^{\mathcal{A}}, s_{3,6}^{\mathcal{A}}\}$, for $n = |U| = 6$, and $m = |S| = 3$, where all edge servers are open. The numbers (indicated by $< \dots >$) next to the users correspond to resource demands in terms of QoV, while the ones next to the servers refer to their computing capacity.

probably possess a finite investment budget for renting the MEC infrastructure (i.e., the BSs here) from the telecommunication carriers [36, 37]. Hence, we introduce another constant, denoted as B, to enforce this *budget* threshold for the SP. Thus, we obtain the following *budget constraint*.

Definition 3 (budget constraint). *$\mathcal{A}$ is considered valid if and only if capacity constraint (2) in not infringed and the following relation holds:*

$$\sum_{j=1}^{J} \beta_j^{\mathcal{A}} \leq B, \tag{3}$$

where $\beta_j^{\mathcal{A}}$ is a binary indicator dictating whether edge server s_j ($1 \leq j \leq m$) is open, i.e, is operational, under the particular strategy; that is, $\beta_j^{\mathcal{A}} = 1$ if yes, whereas $\beta_j^{\mathcal{A}} = 0$ if otherwise.

3.4 Game group interactivity model

Let $d(p, q)$ be a positive-value weight representing the network delay incurred over the link $e(p, q) \in E$, that connects two entities $p, q \in V$. Without any loss of generality, in this paper the weight equates to their in-between euclidean distance, i.e., $d(p, q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$, where (x_p, y_p) and (x_q, y_q) defines the cartesian coordinates of p and q respectively. Note, that in a globe-scale this must be accordingly converted to spherical geolocation coordinates [28].

When an interaction occurs for a pair of entities over multiple links, the total network delay equates to the concatenation of all separate link weights connecting the two [38]. We, thereby, define function $\mathcal{L}^{\mathcal{A}}(p, q) \geq 0$ to capture the total *interaction latency* for any pair $(p, q) \in$

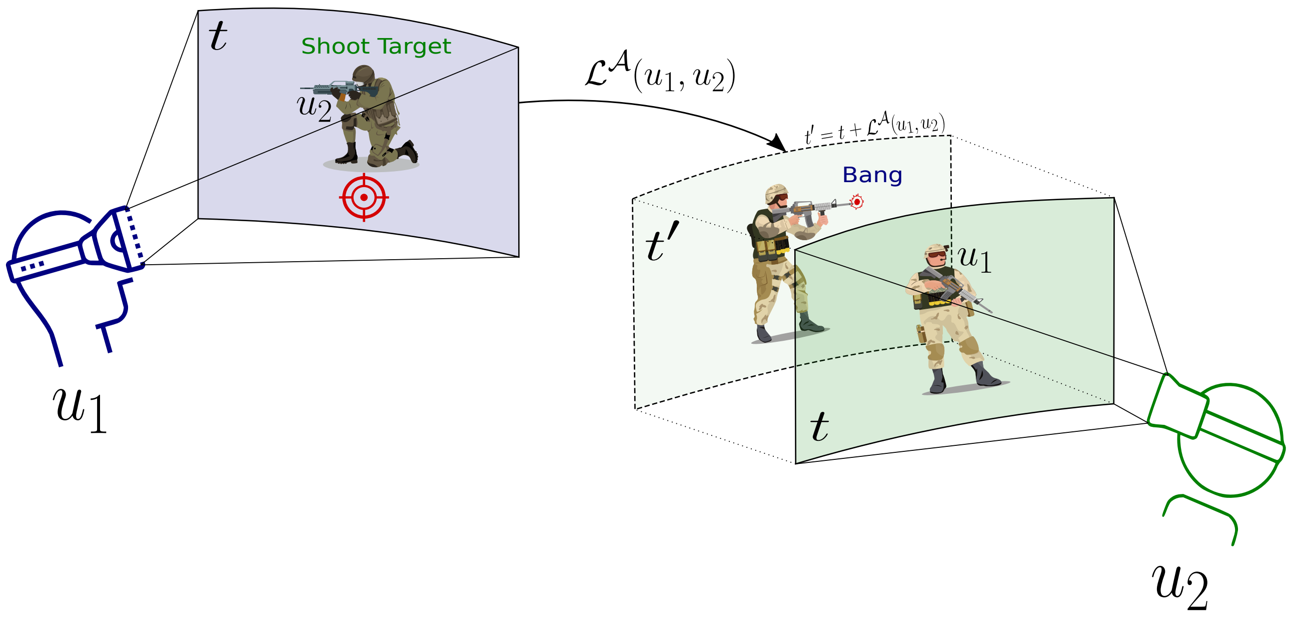

t
Shoot Target
u_2
u_1
$\mathcal{L}^{\mathcal{A}}(u_1, u_2)$
$t' = t + \mathcal{L}^{\mathcal{A}}(u_1, u_2)$
Bang
t'
u_1
t
u_2

Fig. 2 – An example of how interaction latency in VR can favor one user over another.

$V \times V$ under $\mathcal{A}$. Obviously, we then have $\mathcal{L}^{\mathcal{A}}(p, q) = \mathcal{L}^{\mathcal{A}}(q, p)$, while for $p = q$ we get $\mathcal{L}^{\mathcal{A}}(p, q) = 0$.

To visualize the interaction process among users belonging to the same game group, Fig. 1 depicts a small-scale instance of a MEC network for six users and three edge servers, the latter being attached to an equal number of BSs. The coverage range of each BS is also depicted. For u_4, who only interacts with its assigned server s_1, the interaction latency is computed as $\mathcal{L}^{\mathcal{A}}(u_4, s^{\mathcal{A}}_{1,4}) = 2d(u_4, s^{\mathcal{A}}_{1,4})$ which is essentially the Round-Trip Time (RTT). For the interaction of u_2 with u_3, given that u_3 has been ultimately assigned to s_2, because s_3 does not have enough capacity to host both u_3 and u_4, the interaction latency is measured as $\mathcal{L}^{\mathcal{A}}(u_2, u_3) = d(u_2, s^{\mathcal{A}}_{2,2}) + d(s^{\mathcal{A}}_{2,3}, u_3)$. Lastly, for u_5 and u_6, with s_1 and s_3 as their respective servers, the interaction latency becomes $\mathcal{L}^{\mathcal{A}}(u_5, u_6) = d(u_5, s^{\mathcal{A}}_{1,5}) + d(s^{\mathcal{A}}_{1,5}, s^{\mathcal{A}}_{3,6}) + d(s^{\mathcal{A}}_{3,6}, u_6)$.

Generalizing, for any pair $u_p, u_q \in U^k$, associated with $s_w, s_z \in S$ respectively, and engaged with the same game $h_k \in H$, the interaction path from u_p to u_q yields an interaction latency given by:

$$\mathcal{L}^{\mathcal{A}}(u_p, u_q) = d(u_p, s^{\mathcal{A}}_{w,p}) + d(s^{\mathcal{A}}_{w,p}, s^{\mathcal{A}}_{z,q}) + d(s^{\mathcal{A}}_{z,q}, u_q). \quad (4)$$

The same latency applies also for the reverse interaction path, when an operation is initiated from u_q in order to interact with u_p. The reader should note the special case of u_1 in Fig. 1, who is not covered by any BS and unavoidably connects to the remote cloud $\mathcal{R}$, a fact that significantly degrades interactivity compared to the edge servers, due to the long network distance, i.e.,

$$\mathcal{L}^{\mathcal{A}}(u_p, u_q) < d(u_p, \mathcal{R}) + d(\mathcal{R}, s^{\mathcal{A}}_{z,q}) + d(s^{\mathcal{A}}_{z,q}, u_q), \quad (5)$$

$\forall u_p, u_q \in U^k, \forall s_z \in S$.

Even though the described interactivity model does not explicitly account for any user request processing/rendering time on the edge servers, nevertheless, it remains scalable enough in the sense that such conditions can be trivially incorporated without any impact on the quality of our FQEA heuristic solution (presented in later parts of the paper). With that said, in the next section, we proceed with the theoretical formulation of the FQEA problem.

4. FAIRNESS AND QOE FOR VR

As already mentioned, *fairness* is a critical component for multiplayer VR group gaming in order to maintain a compact frame view consistency across all immersed players. Normally, the in-game time of a user lags behind the frame-generation time of its assigned edge server due to the network latency of delivering VR state updates from the server to the user. This is even more profound when the interaction process involves multiple users assigned to different servers, wherein the lagging period is also impacted by inter-server delays. Thus, the interaction latency among players can quickly lead to game unfairness giving the upper hand to some users over their opponents. This may result in the latter losing their loyalty or abandoning the game entirely in search of another, where they will have equal chances in rewards [39]. Moreover, it hurts the total product use and, hence, the SP's possibility for further monetization in subscription-based VR games [13].

To showcase this, assume the VR shooting game depicted in Fig. 2. Two users, that are members of opposing teams, are engaged with the VR game through their respective HMDs. Assume that at some point in time, say t, user u_1 has already targeted its enemy, i.e., user u_2, and has begun shooting. Due to interaction latency between the two users, at time t user u_2 perceives the scene in a different manner. Specifically, u_2 spots u_1 standing idle, giving u_2 plenty of time to engage in combat or take cover, whereas, in reality, u_2 should have taken damage by the bullet of u_1. However, this frame will arrive at u_2 during time $t' = t + \mathcal{L}^{\mathcal{A}}(u_1, u_2)$, a fact that leads to game view inconsistency, and hence unfairness towards u_1 who should have been awarded with a kill.

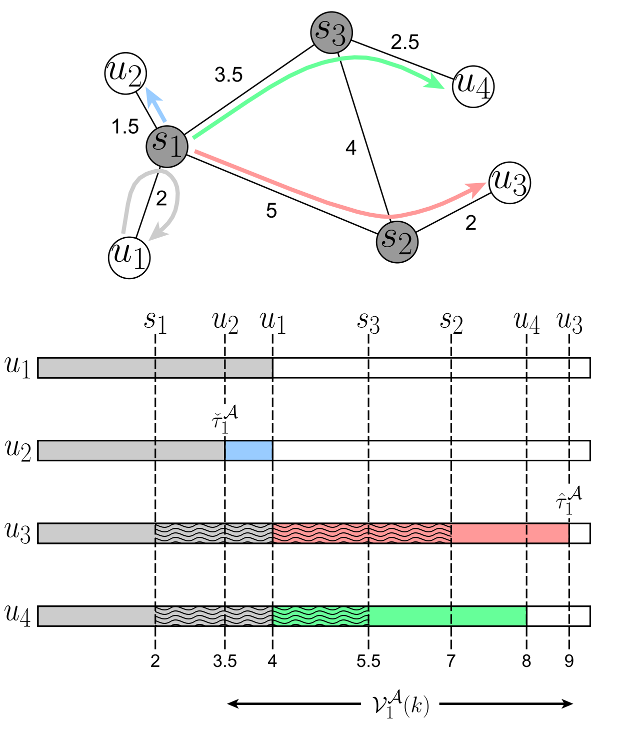
To avoid such view inconsistencies all users should share the same view of the VR application state when their respective in-game time reaches the same value. Since the VR scene state continuously evolves due to both user operations and time passing, to ensure consistency, the generated VR frames that resulted from a user interaction must be displayed to all users belonging to the same game group at the same in-game time. This, in turn, entails the enforcement of a minimum *wait time* after each user interaction issuance.

4.1 View inconsistency analysis

Assuming that the SP can guarantee the timely synchronization of all involved parties (synchronization is out of the scope of this paper because such schemes are already widely used in large-scale virtual environments to process data in correct temporal order [11]), here we analyze the minimum wait time required to address the interaction latency variance witnessed by each user in a VR game, say $h_k \in H$.

In view of Eq. *(4)*, for some $u_i \in U^k$ the scene view progress of the h_k's application state should be measured by the time elapsed since the user's input command and the moment its rendered update is displayed by the devices of all $u_n \in U^k$ engaged with the same VR application (including the device of u_i). However, this time varies among the different users, especially those assigned to different edge servers, since the interaction latency introduces unavoidable lag due to inter-server and user-server delays. Let $\pi^k_{i,n}$ denote the *lag difference* experienced between the interaction of user u_i and some other user u_n, where $u_i, u_n \in U^k$, i.e., $\pi^k_{i,n} = \mathcal{L}^{\mathcal{A}}(u_i, u_i) - \mathcal{L}^{\mathcal{A}}(u_i, u_n)$. A positive offset in the lag means that the new update, which resulted from the operation of u_i, is delivered to u_n ahead of u_i, whereas a negative one means that u_n's time of update reception falls behind that of u_i.

Take for example the small-scale graph in Fig. 3, where the interaction process for four users belonging to the same subset U^k is depicted considering the case where an in-game interaction is initiated by u_1. According to our former analysis, for u_1 the interaction latency is equal to the RTT, i.e., $\mathcal{L}^{\mathcal{A}}(u_1, u_1) = 4$. For u_2, who is assigned to the same server, i.e., the s_1, but positioned closer than u_1 (in terms of network delay), the $\mathcal{L}^{\mathcal{A}}(u_1, u_2) = 3.5$. Therefore, their in-between lag difference equates to $\pi^k_{1,2} = 0.5$. As for u_1's lag difference with users u_2 and u_3, we have



s_3
2.5
u_2
3.5
u_4
1.5
s_1
4
u_3
2
5
2
u_1
s_2
s_1 u_2 u_1 s_3 s_2 u_4 u_3
u_1
$\check{\tau}_1^{\mathcal{A}}$
u_2
$\hat{\tau}_1^{\mathcal{A}}$
u_3
u_4
2 3.5 4 5.5 7 8 9
$\mathcal{V}_1^{\mathcal{A}}(k)$

Fig. 3 – Lag variance among a set of four users u_i $(i = 1, \dots, 4)$ playing game h_k, when an interaction is initiated by user u_1. On the top side the formed graph, where the colored arrows represent the different interaction paths and the link numbers correspond to network delay weights; on the bottom side, the inter-player interaction process in time (depicted in a horizontal fashion) with corresponding colors, where the wave overlay pattern depicts inter-server delays and the numbers refer to points in time according to the network delay weights.

$\pi^k_{1,2} = -5$ and $\pi^k_{1,2} = -4$ respectively.

This clearly exemplifies that the achievable consistency of game scene viewing among users of the same VR game group (i.e., those immersed in h_k), when an operation is initiated by $u_i \in U^k$, is upper-bounded by the maximum of absolute interaction lag difference among all involved user pairs (u_i, u_n), that is:

$$|\pi^k_{i,n}| \leq |\max \mathcal{L}^{\mathcal{A}}(u_i, u_x) - \min \mathcal{L}^{\mathcal{A}}(u_i, u_y)|, \qquad (6)$$

$\forall u_x, u_y \in U^k$.

For brevity, denote as $\hat{\tau}_i^{\mathcal{A}} = \max \mathcal{L}^{\mathcal{A}}(u_i, u_x)_{\forall u_x \in U^k}$ and as $\check{\tau}_i^{\mathcal{A}} = \min \mathcal{L}^{\mathcal{A}}(u_i, u_y)_{\forall u_y \in U^k}$ the maximum and minimum interaction latency respectively, experienced by users playing h_k under $\mathcal{A}$, when the source of interaction is u_i. Then, in view of Ineq. *(6)*, for users in a VR application to simultaneously witness the same view changes, the client of each $u_n \in U^k$ must enforce a *wait-time* period of $t^k_n(i) = |(\hat{\tau}_i^{\mathcal{A}} - \check{\tau}_i^{\mathcal{A}}) + (\mathcal{L}^{\mathcal{A}}(u_i, u_n) - \hat{\tau}_i^{\mathcal{A}})| \leq |\hat{\tau}_i^{\mathcal{A}} - \check{\tau}_i^{\mathcal{A}}|$, before displaying the newly rendered frame that resulted from the operation of $u_i \in U^k$, including the case for $n = i$. Ergo, maximizing fairness for u_i is equivalent to minimizing the maximum lag difference with all its user counterparts engaged with the same VR game.

Denote as $\mathcal{V}_i^{\mathcal{A}}(k)$ the *view inconsistency* of $u_i \in U^k$ under

$\mathcal{A}$, expressed as the *maximum lag variance* experienced by users immersed in the VR game h_k, when the source of interaction is u_i, i.e.,

$$\mathcal{V}_i^{\mathcal{A}}(k) = |\hat{\tau}_i^{\mathcal{A}} - \check{\tau}_i^{\mathcal{A}}|. \quad (7)$$

In Fig. 3 for instance, we have $\mathcal{V}_1^{\mathcal{A}}(k) = |\hat{\tau}_1^{\mathcal{A}} - \check{\tau}_1^{\mathcal{A}}| = |\mathcal{L}^{\mathcal{A}}(u_1, u_3) - \mathcal{L}^{\mathcal{A}}(u_1, u_2)| = 5.5$. Thus, to reach view consistency for the interaction of u_1, we get $t_1^k(1) = 5$, $t_2^k(1) = 5.5$, $t_3^k(1) = 0$, and $t_4^k(1) = 1$.

Clearly, to offer a truly immersive experience in social VR, the wait-time for each user must be kept as low as possible. This embodies the reduction in view inconsistencies. Alas, due to the variability in inter-server delays, different results are obtained if an interaction is initiated by some other user $u_{m,m\neq i} \in U^k$, and so $\mathcal{V}_i^{\mathcal{A}}(k)$ $(\forall i, k)$ is an important QoS factor, that must be taken heavily into consideration along with QoV, in defining the overall QoE of the users engaged with the VR applications.

4.2 QoS-QoE correlation

The QoE of the users tightly depends on their delivered QoS. Generally speaking, this correlation is application-specific but aligns well with video streaming services [40]. For example, in cloud games, it has been shown that QoE is affected by the QoS in terms of the number of virtual machines instantiated [41] or the amount of rendering resources allocated [42]. Similarly, for edge computing environments, the research in [22] reveals that a user's QoE relies on the computing resources it receives from an edge server. As the user's QoS level increases, so does the perceived QoE. Yet, the QoS-QoE curve is not linear. Instead, it is broadly acknowledged in several relevant studies (e.g., [43]), that the relationship follows a sigmoid function, rising slowly at first, then speeding up, before finally converging.

To exemplify this in a VR scenario, consider a game that is played on a smartphone. For the player's eyes, the difference in graphics resolution from 240p to 360p is noticeable but not nearly enough to increase its QoE significantly. Contrary to this, when it becomes 720p, or even more so when it reaches 1080p or 2k, the difference, at the cost perhaps of a slightly more computational expense, is substantial and so its QoE rises drastically. From thereon, if the resolution increases further, e.g., to 4k, the difference becomes almost undetectable by the player's eyes and so its QoE barely increases (if at all). The reverse attitude is expected as fairness degrades among the users immersed in the same VR game. Thus, the sigmoid correlation between QoS and QoE fits well to our model.

In this research, the QoS of a VR user is jointly determined by two elements. First, by its individual QoV, which can be improved by provisioning additional edge computing resources for increased graphics computation and frame rendering. Second, by the achieved fairness in regards to view inconsistency between the user and its game group.

To quantify the QoE, similar to [23], we choose the logistic function, as a generalized expression of the sigmoidal correlation with QoS. Formally, it is formulated as:

$$\varepsilon_i^{\mathcal{A}} = \frac{Q}{1 + e^{\omega_1 (q_i^{\mathcal{A}} - \omega_2)}}, \quad (8)$$

where $\varepsilon_i^{\mathcal{A}}$ denotes the QoE level that is perceived by user u_i given its QoS $q_i^{\mathcal{A}}$, Q is the maximum possible QoE level, whereas ω_1 and ω_2 are model parameters that tune the correlation, with the former controlling the growth of the QoE curve while the latter representing its midpoint. Note that $q_i^{\mathcal{A}}$ is measured by the ratio of view inconsistency experienced by u_i to the actual QoV in terms of computing resources allocated by its assigned edge server, i.e.,

$$q_i^{\mathcal{A}} = \sum_{k=1}^{K} \frac{\mathcal{V}_i^{\mathcal{A}}(k)}{l_i^{\mathcal{A}}} z_i^k, \quad (9)$$

where $l_i^{\mathcal{A}}$ denotes the gained QoV level of u_i under $\mathcal{A}$ and $z_i^k \in \{0, 1\}$ is an index expressing whether $u_i \in U$ is *engaged* with the VR game $h_k \in H$.

4.3 Problem formulation

In view of Eq. *(8)*, we introduce here the FQEA problem. Recall that we primarily address the problem from the viewpoint of the users.

Definition 4 (FQEA). *Given a MEC network $G = (V, E)$ where V contains a set of servers S and a set of users U, and the network interaction delay weight $d(p,q) > 0$ $(p, q \in V)$ for each link $e(p,q) \in E$, the objective of FQEA is to find a valid edge allocation strategy $\mathcal{A}$ that maximizes the users' average QoE, subject to the proximity, capacity and budget constraints of the MEC, or equivalently:*

$$\textit{o.f.:} \quad \max_{\mathcal{A}} \frac{1}{I} \sum_{j=1}^{J} \sum_{i=1}^{I} \varepsilon_i^{\mathcal{A}} \alpha_{i,j}^{\mathcal{A}} \quad (10)\text{a}$$

$$\textit{s.t.:} \quad \alpha_{i,j}^{\mathcal{A}} = 0 \quad \forall i, j \in \{i, j | \lambda_{i,j} > \varrho_j\}, \quad (10)\text{b}$$

$$\sum_{j=1}^{J} \alpha_{i,j}^{\mathcal{A}} = 1, \quad \forall i \quad (10)\text{c}$$

$$\sum_{j=1}^{J} \beta_j^{\mathcal{A}} \leq B, \quad (10)\text{d}$$

$$\sum_{i=1}^{I} r_i \alpha_{i,j}^{\mathcal{A}} \leq C_j \quad \forall j, \quad (10)\text{e}$$

$$\alpha_{i,j}^{\mathcal{A}} \leq \beta_j^{\mathcal{A}} \quad \forall i, j \quad (10)\text{f}$$

$$\alpha_{i,j}^{\mathcal{A}}, \beta_j^{\mathcal{A}} \in \{0, 1\} \quad \forall i, j. \quad (10)\text{g}$$

In the above model, the objective function *(10)*a maximizes the average QoE of the immersed users. Constraint *(10)*b validates that each edge server can serve only users that are located within its BS's coverage range. Binding constraint *(10)*c restricts the assignment of each user to at most one edge server. Constraint *(10)*d elicits that the sum of open edge servers cannot surpass the

SP's available edge investment budget. Constraint *(10)*e guarantees that the total amount of resource demands allocated to all users assigned to a particular edge server will not exceed its computing capacity. Constraint *(10)*f enforces that for a given edge server to serve a user, it must first be open. Finally, the obligatory conditions for the binary variables are given in constraint *(10)*g. All coefficients are assumed to be non-negative and integral.

Note, that in this paper we intentionally measure fairness with respect to view inconsistency caused by the lag variance of users assigned just to the MEC system, and not to $\mathcal{R}$, since the latter, as already mentioned, by definition incurs unacceptable interactivity. Likewise, if $\alpha_{i,j}^{\mathcal{A}} = 0, \forall j$, then we let $\varepsilon_i^{\mathcal{A}} = 0$, because users assigned to $\mathcal{R}$ are also assumed to experience unacceptable QoE.

4.4 NP-hardness

Unfortunately, as it turns out, finding an optimal allocation strategy to FQEA, that fulfills all aforementioned criteria, can be extremely challenging. This is proven with the following theorem.

Theorem 1. *The FQEA problem is NP-hard.*

Proof. To prove the NP-hardness of FQEA, we first introduce the classical Generalized Assignment Problem (GAP) [44]. In the GAP, we are given a set of items $N = \{1, \dots, n\}$ and a set of bins (or knapsacks) $M = \{1, \dots, m\}$ with positive and possibly different capacities f_j ($j = 1, \dots, m$). Each item, depending on the bin, is characterized by a profit $p_{i,j}$ and a cost weight $w_{i,j}$ ($i = 1, \dots, n$). Then, we seek to assign each item to at most one bin ensuring that none of the capacity constraints are violated. The objective is to select a feasible solution, such that the total profit of the bins is maximized.

Next, we show that FQEA can be reduced from GAP. The reduction is as follows. First, we relax constraint *(10)*b of FQEA, allowing users to be assigned to any edge server, if possible. Second, we set $B = |S|$. Then, given an instance *GAP*$(N, M, f_j, w_{i,j}, p_i)$, we can construct in polynomial time an instance *FQEA*$(U, S, C_j, r_i, \varepsilon_i^{\mathcal{A}})$, with the objective of maximizing QoE. Note that in FQEA, users unable to connect to the MEC network are redirected to the remote cloud with zero QoE. Thus, we include $\mathcal{R}$ as part of the solution, by setting $S = J + 1$ and projecting S back to M. In this way, *FQEA* now perfectly mirrors *GAP*. Ergo, FQEA is indeed reducible from GAP. Because the latter is known to be NP-hard [44], we conclude that the former as a special case is also NP-hard, and the theorem immediately holds. □

5. ALGORITHM DESIGN

Due to the nature of FQEA, finding optimal solutions in large-scale multiplayer VR games would prove intractable even with vast computing resources available, since any brute-force search will result in exponential time complexity. This motivates the design of a scalable solution. Next, we propose our heuristic allocation algorithm, termed as the *"Fairness and QoE-Based Edge Allocation Heuristic"* (FQEA-H) algorithm. Its pseudo code is presented in Algorithm 1.

Algorithm 1 FQEA-H

```
Input: $G = (V, E), U, S, H$;
Output: User-to-Edge Server allocation $\mathcal{A}$;
1: Sort $U$ in ascending order of resource demands;
2: EdgeAssignemnt;
3: QoEImprovement;
4: return $\mathcal{A}$;
```

Let $S^o \subseteq S$ define the subset of *open* edge servers, i.e., those that host users and, thus, are operational. Denote as S_{u_i} the subset of edge servers s_j, $\forall j = \{1, \dots, J\}$, where $\lambda_{i,j} \leq \varrho_j$ holds, i.e., the subset of *accessible* edge servers by user u_i. Then, $S_{u_i}^o \subseteq S^o \cap S_{u_i}$ is the subset of u_i' *neighbor open* edge servers. Also, denote as $c_j \leq C_j$ the residual capacity of edge server s_j, i.e., the amount of its idle computing resources that is not currently in use. FQEA-H consists of two main phases, before outputting the allocation strategy $\mathcal{A}$. The description of the algorithm's execution is as follows.

Prior to assignment initiation, all incoming users are sorted and scheduled in increasing order of their resource demands (Line 1), which are measured according to the recommended display requirements of their VR device or their personal gaming preferences in order to achieve the minimum desired QoV. This means that users with lesser demands are prioritized, avoiding early over-capacity issues, that could manifest at certain edge servers during the early stages of the assignment phase, and preventing the allocation of any future users.

5.1 Edge assignment phase

Post sorting the users, the *"Edge Assignment Phase"* kicks off (i.e., Line 2 of Algorithm 1), whereby the users are checked sequentially to determine for each one the most favorable neighbor edge server for assignment. The details of this phase are included in Algorithm 2. Specifically, each user $u_i \in U$ scans its neighborhood S_{u_i} (Line 3) for potential edge servers that meet proximity constraint *(1)*; that is, the user's geographical location must lie within the boundaries of the corresponding BSs' coverage range.

When u_i has discovered all candidate servers that meet the above criterion, it traverses them to verify whether there exists any whose capacity constraint *(2)* is not violated (Line 4). If the verification proves successful, then it computes for each candidate server the ratio of its experienced view inconsistency, with all other users who belong to the same $h_k \in H$ and have already been admitted to edge servers in a previous step, to the residual capacity of the server (Lines 6-12). This is done for two reasons.

Algorithm 2 EdgeAssignemnt

```
Input: $G = (V, E), U, S, H$, EXISTS;
 1: for each $u_i \in U$ do
 2:     EXISTS== $FALSE$;
 3:     for each $s_j \in S_{u_i}$ do
 4:         if $r_i + \sum_{n=1,n\neq i}^{I} r_n \alpha_{n,j}^{\mathcal{A}} \leq C_j$ then
 5:             EXISTS= $TRUE$;
 6:             if $s_j \in S_{u_i}^o$ then
 7:                 Compute $\frac{\mathcal{V}_i^{\mathcal{A}}(k)}{c_j}$;
 8:             else
 9:                 if $S^o \cup \{s_j\} \leq B$ then
10:                     Compute $\frac{\mathcal{V}_i^{\mathcal{A}}(k)}{c_j}$;
11:                 end if
12:             end if
13:         end if
14:     end for
15:     if EXISTS== $TRUE$ then
16:         Assign $u_i$ to $s_j : j \leftarrow \arg\min_{\mathcal{A}} \left\{ \frac{\mathcal{V}_i^{\mathcal{A}}(k)}{c_j} \right\}$;
17:         if $s_j \notin S_{u_i}^o$ then
18:             $S^o \leftarrow S^o \cup \{s_j\}$;
19:         end if
20:     else
21:         Assign $u_i$ to the remote cloud infrastructure $\mathcal{R}$;
22:     end if
23: end for
```

First, to discover the location that optimizes fairness after user u_i is allocated. Second, to find the server that has potentially enough space to either accommodate future users of the same game as u_i or maximize u_i's own QoE. After measuring all ratios, u_i is ultimately assigned to the s_j that yields the minimum one (Line 16).

The reader should note that, if the examined $s_j \in S_{u_i}$ does not belong to the set of already *open* servers (i.e, $s_j \notin S_{u_i}^o$), a validation test is conducted by the algorithm to certify that the budget constraint *(3)* is not infringed for the SP with the addition of s_j (Lines 9-11), in the case the latter is indeed selected for the particular assignment. In other words, FQEA-H checks whether $S^o \cup \{s_j\} \leq B$ before proceeding with the lag variance calculation. If the condition fails, then the server is skipped to maintain the number of open servers within the acceptable budget threshold.

Finally, if the selected edge server has just been opened, it is now included to S^o (Lines 17-19). Otherwise, if no server has been discovered that meets all necessary criteria, u_i is inevitably redirected to $\mathcal{R}$ (Line 21), which is generally not desirable due to the high interaction latency and long network distances, that incur high lag variance, significantly hurting fairness and, hence, QoE.

The aforementioned process repeats until all users have been assigned either to edge servers or the remote cloud. At that point, the edge assignment phase terminates and the QoE improvement phase begins.

5.2 QoE improvement phase

During the second phase of FQEA-H, namely the *"QoE Improvement Phase"* (i.e., Line 3 of Algorithm 1), the QoE

Algorithm 3 QoEImprovement

```
Input: $G = (V, E), U, S, H$, IMPROVED;
 1: IMPROVED= $TRUE$;
 2: while IMPROVED== $TRUE$ do
 3:     IMPROVED= $FALSE$;
 4:     for each $u_i \in U$ do
 5:         if $\sum_{n=1}^{I} r_n \alpha_{n,j}^{\mathcal{A}} < C_j$ then
 6:             Attempt to increase the $u_i$'s QoV by one level;
 7:             if Successful then
 8:                 $l_i^{\mathcal{A}} \leftarrow l_i^{\mathcal{A}} + 1$;
 9:                 IMPROVED= $TRUE$;
10:             end if
11:         end if
12:     end for
13: end while
```

of the edge-assigned users is iteratively improved by sequentially attempting to increase their QoS in terms of QoV levels. The steps of this phase are captured within Algorithm 3.

In particular, for each user $u_i \in U$ (Line 4), given that its associated edge server can support higher QoV, we increase its previous QoV by one level (Line 6). In other words, if the resource demands required for a QoV enhancement do not exhaust the capacity of the edge server s_j, where $\alpha_{i,j}^{\mathcal{A}} = 1$ (i.e., Line 5), then a QoV level improvement takes place (Line 8).

When all users have been investigated, if the QoV for some user(s) is indeed increased (Line 9), then a new round is initiated to check whether further improvements on QoE can happen. This process is enclosed in the **While** loop (Line 2), and repeats until no more augmentations can occur, e.g., due to server capacity violations, or until the QoV of all users reaches the maximum supported level by the SP, at which point the algorithm exits the loop (Line 13) and FQEA-H returns the generated allocation strategy $\mathcal{A}$ (i.e., Line 4 of Algorithm 1).

5.3 Time complexity analysis

Because users are being checked independently, FQEA-H avoids circumstances where a simultaneous assignment of multiple users could result in over-utilization of the edge servers and overloading issues. As the solution space is finite, FQEA-H also terminates its edge assignment phase after a finite number of steps, which are upper-bounded by the number of users to be assigned, i.e., by $I = |U|$. In addition, at each new iteration, that necessitates the opening of a new edge server, the budget constraint is first validated to guarantee that no infringements manifest. As the number of edge servers is also finite, and the users that are unable to be admitted to the MEC (e.g., due to proximity constraints) can instead connect to the remote cloud infrastructure $\mathcal{R}$, it is concluded that the particular phase will converge to a valid assignment.

In regards to the QoE improvement phase, its **While** loop will run repeatedly for multiple rounds, until no more improvements can occur. Denote as $\mathcal{X}$ the number of rounds required. We know that $\mathcal{X}$ is upper-bounded by the maximum number of available QoV levels offered by the SP. Considering that the number of assigned users that resulted from the assignment phase is also finite, as well as the limited capacity of the opened edge servers, it is concluded that the particular phase will terminate after at most $\mathcal{X}$ rounds. In summary, it is guaranteed that FQEA-H, despite being comprised of two independent phases, will converge to a solution after a finite number of steps.

In view of these findings, we can prove the time complexity of the proposed algorithm with the following theorem.

Theorem 2. *The time complexity of FQEA-H is $\mathcal{O}(|U|^2|S|)$.*

Proof. Initially, all users are sorted based on their resource demands, a process that leads to a maximum of $\mathcal{O}(|U|\log|U|)$ time complexity. We now investigate the two phases of FQEA-H separately. i) For the first phase, each scheduled user $u_i \in U^k \subseteq U$, where $k = \{1, \dots, K\}$, must transverse its accessible edge servers $s_j \in S_{u_i} \subseteq S$ to determine the most suitable assignment location that will result in the minimum $\frac{\mathcal{V}_i^{\mathcal{A}}(k)}{c_j}$. This process introduces a maximum time complexity of $\mathcal{O}(|U||S|)$ when all users can be assigned to any server due to proximity and budget constraints' relaxation. However, measuring these ratios warrants the computation of each $\mathcal{L}^{\mathcal{A}}(u_i, u_n)_{\forall u_n \in U^k}$, which for the extreme scenario, when all users are also engaged in a single game $h_k \in H$, results in a maximum time complexity of $\mathcal{O}(|U|^2)$ since there exist $|U| \times |U|$ possible user pairs as (u_i, u_n), $\forall i, n$, including the case where $i = n$. ii) The second phase is dominated by the **While** loop, wherein all users assigned to edge servers are examined sequentially to test whether their QoE can be augmented by improving their QoS with respect to their QoV levels. Considering a perfect user assignment of 100% edge admission, the worst-case time for one round of the loop equates to $\mathcal{O}(|U|)$. However, this process runs iteratively for at most X rounds. Therefore, the overall time-complexity becomes $\mathcal{O}(\mathcal{X}|U|)$. Based on the preceding, we conclude that the total worst-case time complexity of FQEA-H is $\mathcal{O}(|U|\log|U| + |U|^2|S| + \mathcal{X}|U|)$, or equivalently $\mathcal{O}(|U|^2|S|)$. This completes the proof. □

It is worth reminding that FQEA constitutes a multi-variant multi-constrained optimization problem, synthesized by a combination of different facets, i.e., the maximization of user edge admission, the minimization of user-server delay, the maximization of inter-player interactivity with minimum lag variance as well as, finally, the maximization of QoS/QoE. Ergo, FQEA-H cannot guarantee the discovery of the optimal solution because a plethora of pivotal factors must be weighted-in during the optimization to reach a sensitive trade-off. Nevertheless, its execution is ensured to converge towards a suitable allocation strategy regardless of the number of incoming users or available edge servers.

Besides, our approach proposed in this research can also be adapted to solve the dynamic FQEA version. Although, here, we study the problem in quasi-static mode, rather than explicitly taking into account user mobility or demand variability, FQEA-H can be iterated over time to address such issues, e.g., by splitting the long-term FQEA into short-term time slots, and then employing FQEA-H to solve them individually in each time slot. However, this is beyond the scope of the current paper and bears further analysis and future investigation.

Table 2 – QoV Levels, used in the evaluation, based on the resources allocated $\forall u_i \in U$.

Graphics Resolution	Resources	QoV Level
360p	5	1
720p	7	2
1080p	9	3
2k	11	4
4k	13	5

6. EVALUATION

To evaluate the performance of FQEA-H, using the OMNeT++ v5.6.2[1], we conduct trace-driven simulations on the EUA dataset[2], which has been broadly utilized for research purposes in edge computing [45].

6.1 Dataset and network configuration

The EUA dataset enlists the geolocations of various cellular BSs in Australia from a plethora of telecommunication carriers. For our experiments, we select a fraction of the total BSs, i.e., 125 BSs that are situated in the Central Business District (CBD) of Melbourne. CBD covers an area of approximately $6.2\ km^2$. Unless otherwise stated, after normalizing this area into a topology (1×1), $|U| = I = 1000$ users are randomly placed in a uniform manner around the BSs. We, also, assume that each BS corresponds to the location of one edge server. Thus, we have at most $|S| = J = 125$ edge servers, whereas for the budget limit, unless otherwise specified, it is set in all cases as $B = 100$.

The users form links with the edge servers based on their BSs' coverage range which, if not defined explicitly, is randomly set to $\varrho_j \in [0.3, 0.4]$ $(0 \leq j \leq J)$, to mirror cellular towers having different communication power. For the edge servers, the available computing capacity, unless otherwise stated, is randomized within $C_j \in [80, 100]$. As in [16], the resource demands of each user u_i are randomly drawn from a pool of five values, i.e., $r_i \in \{5, 7, 9, 11, 13\}$ $(0 \leq i \leq I)$, to capture the minimum needs for different QoV levels, as detailed in Table 2. Meanwhile, for the QoE metric, we additionally set $Q = 5$, $\omega_1 = 3$ and $\omega_2 = 1$, according to [22]. For users allocated to $\mathcal{R}$ we set their QoE equal to zero. Finally, each u_i randomly selects a VR game application from a set of four available, i.e., $|H| = K = 4$ here. Note, that all provided

[1]OMNeT++ Discrete Event Simulator: https://omnetpp.org/
[2]EUA Datasets: https://github.com/swinedge/eua-dataset

parameter values refer to representative cases. In a real-world scenario, the SPs can customize these attributes according to VR application-specific requirements.

6.2 Simulation scenarios and benchmarks

Based on the preceding assumptions, five sets of simulations are conducted.

#1: **Varying number of edge servers**: A certain fraction of the total 125 edge servers are randomly selected as $|S| = \{75, 85, \dots, 125\}$.

#2: **Varying computing capacity**: The edge servers' capacity is steadily amplified as $C_j = \{80, 90, \dots, 130\}$.

#3: **Varying number of users**: The number of users is sequentially increased as $|U| = \{700, 800, \dots, 1200\}$.

#4: **Varying coverage range**: The coverage range of each BS is gradually expanded as $\varrho_j = \{0.15, 0.2, \dots, 0.4\}$.

#5: **Varying budget**: The maximum number of edge servers that are allowed to be open simultaneously, is incrementally raised as $B = \{85, 90, \dots, 110\}$.

For each of these scenarios, FQEA-H is compared against five benchmark algorithms under the same conditions:

- *Distributed Greedy Minimum Delay* (DGMIN) [26]: This is a greedy approach for optimizing interactivity. The algorithm connects newly joint users by computing, under all possible server assignments, the maximum interaction path length with all previously allocated users. Then it selects the server that leads to the minimum maximum interaction path length among all eligible servers to optimize the maximum lag variance.
- *QoE-Aware Edge User Allocation* (QoEUA) [22]: This is a state-of-the-art allocation algorithm, where users are first sorted in increasing order of neighborhood size, and then are repeatedly assigned to the edge servers with the most idle computing resources to increase their QoV. During this process, the users can switch servers via reallocation until their QoE is maximized or the capacity is exhausted.
- *Most Capacity First* (MCF) [16]: This is a state-of-the-art approach, that connects a sorted set of users in increasing order of their resource demands, prioritizing the active edge servers. Its goal is to maximize user assignments and minimize the number of required servers by intelligent resource allocation.
- *Nearest Server* (NS) [38]: This is a classic naive approach, where users always select the nearest server, capable of serving their demands, to minimize user-server delays. However, the algorithm remains oblivious to any inter-server delays that affect fairness.
- *Random Server* (RND): This is a baseline algorithm, where users are randomly assigned to servers by continuously seeking the ones that can accommodate their resource demands.

All benchmarks are adapted here to satisfy the FQEA constraints. Besides, to increase the validity of the comparison, ten independent runs are executed for each scenario, and the results are then averaged and captured within the following plots. The reader should also note, at this point, that the benchmark algorithms are carefully chosen from literature, as each one deals with one or more facets of the FQEA problem, as discussed earlier. In addition, they have been shown to generate near-optimal solutions to the corresponding allocation problems, a fact that justifies their suitability for comparison purposes.

6.3 Numerical results

We now present the numerical results obtained by the simulations that compare the effectiveness of all algorithms. The comparison is conducted on the basis of the following performance metrics: i) The *admission rate*, denoted as $U_S^{\mathcal{A}}$ and measured by the percentage of users allocated to edge servers, i.e., $U_S^{\mathcal{A}} = \frac{1}{I}\sum_{j=1}^{J}\sum_{i=1}^{I}\alpha_{i,j}^{\mathcal{A}}100\%$; higher values are better. ii) The *fairness loss*, denoted as $\mathcal{F}^{\mathcal{A}}$ and computed as the average sum of view inconsistencies experienced by all users assigned to the MEC for all hosted VR game applications, i.e, $\mathcal{F}^{\mathcal{A}} = \frac{1}{|U_S^{\mathcal{A}}|}\sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i=1}^{I}\mathcal{V}_i^{\mathcal{A}}(k)z_i^k\alpha_{i,j}^{\mathcal{A}}$; lower values are better. Finally, iii) the users' *average perceived QoE*, i.e., FQEA's objective function, which for convenience is denoted as $\mathcal{E}^{\mathcal{A}}$ and compiled as $\mathcal{E}^{\mathcal{A}} = \frac{1}{I}\sum_{j=1}^{J}\sum_{i=1}^{I}\varepsilon_i^{\mathcal{A}}\alpha_{i,j}^{\mathcal{A}}$; higher values are better.

6.3.1 Varying number of servers

Fig. 4 encapsulates results from Scenario #1, where we vary the number of edge servers. Beginning with the impact on admission, we witness from Fig. 4(a) a positive correlation between the number of available edge servers ($|S|$) and the percentage of assigned users for all algorithms. Obviously, as the number of servers rises so does the admission rate for all approaches. However, for lower values, FQEA-H achieves higher admission rates than most alternatives, which are only slightly surpassed by MCF, which clearly aims at user assignment maximization. For $|S| \geq 105$ the results of all algorithms begin to converge and closely approximate one another at around $U_S^{\mathcal{A}} = 96\%$, with QoEUA and MCF performing the best with marginal, however, difference from our algorithm. In regards to fairness loss, we can observe from Fig. 4(b) that for $|S| \leq 95$ all algorithms follow a similar pattern, with $\mathcal{F}^{\mathcal{A}}$ growing as $|S|$ increases, which is to be expected since the low number of servers incurs higher inter-server delays for the growing number of admitted users. Still, FQEA-H manages to outperform the bench-

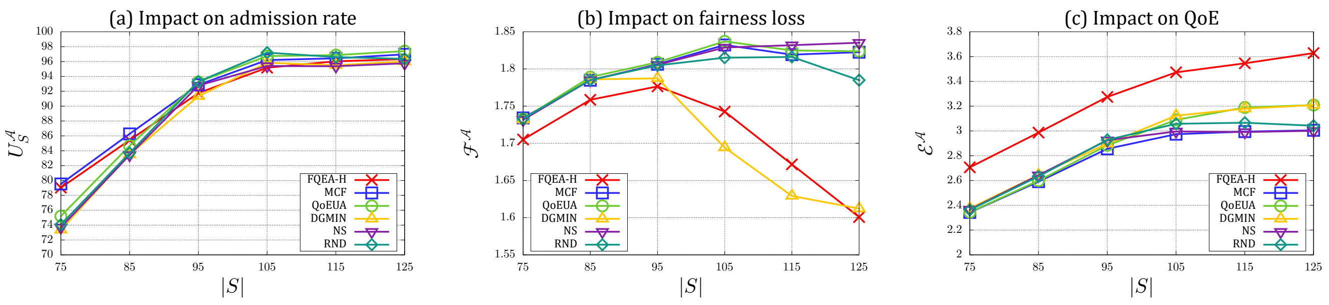


Fig. 4 – (a) User admission, (b) fairness loss, and (c) average QoE as a function of the edge servers' number.

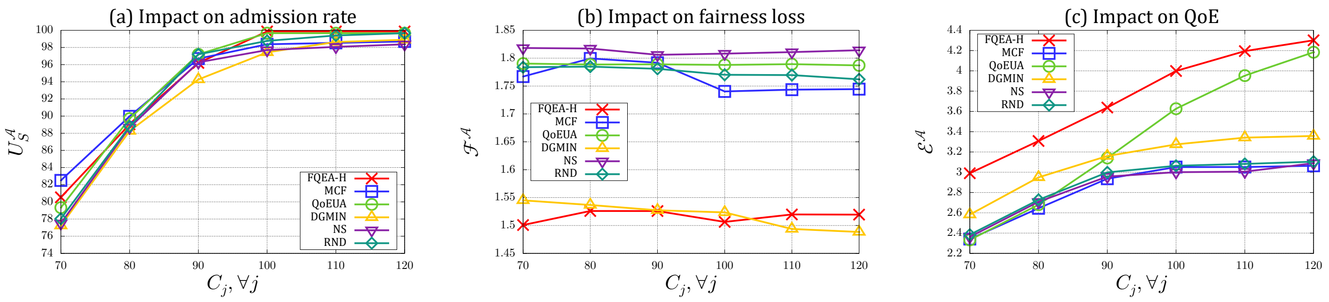


Fig. 5 – (a) User admission, (b) fairness loss, and (c) average QoE as a function of the edge servers' capacity.

marks, including DGMIN whose pure goal is the minimization of the fairness loss. From thereon, enough servers become available for these two algorithms to begin effectively optimizing fairness and significantly outmatch the rest, who present a far worse behavior. Thus, $\mathcal{F}^{\mathcal{A}}$ rapidly decays, with DGMIN experiencing the fastest initial decreasing rate, which begins to converge for $|S| \geq 115$, while for FQEA-H it continues to drop sharply, leading for $|S| = 125$ to FQEA-H surpassing DGMIN once more. Finally, in terms of average QoE, i.e., FQEA's objective, it is evident from Fig. 4(c) that FQEA-H dominates in all cases, managing a substantial performance gap of at least 13% over all other alternatives, with QoEUA and DGMIN falling far behind, while the remainder algorithms performing even worse. In fact, the gap in $\mathcal{E}^{\mathcal{A}}$ widens further as $|S|$ increases, making FQEA-H the best candidate for maximizing QoE.

6.3.2 *Varying computing capacity*

Fig. 5 plots results obtained by Scenario #2. In this case, we vary the amount of available computing capacity while fixing the $|S| = 125$. From Fig. 5(a) we can remark that the proposed FQEA-H again performs exceedingly well in terms of $U_S^{\mathcal{A}}$, making the most out of the available edge resources. Its performance, when the $C_j \leq 90$, is only surpassed by MCF and on occasion slightly by QoEUA, whereas for $C_j \geq 100$, FQEA-H achieves the best edge admission rate, close to 100%, with the worst overall performance being exhibited by DGMIN and NS. Generally, the capacity increase has a positive impact on all algorithms, allowing for more users to be assigned to edge servers. However, as corroborated in Fig. 5(b), with the exception of DGMIN and FQEA-H, all algorithms perform poorly in regards to fairness loss. Clearly, the worst behavior is introduced by NS. This is in accordance with past literature, that wants nearest-server assignments to be far from optimal, and verifies the claim that considering only user-server interaction delays is not enough when addressing view consistency and, hence, fairness. The same apply to QoEUA, MCF and RND. In contrast, DGMIN and FQEA-H perform equally well, a clear testament to their capability to efficiently administrate the servers' computing resources in order to minimize the $\mathcal{F}^{\mathcal{A}}$ with a performance margin ranging from 14% to 22% with all other benchmarks. Noteworthy is also the fact that, for the most part, FQEA-H slightly surpasses DGMIN. Nevertheless, the clear advantage of our algorithm is depicted in Fig. 5(c), wherein we can see that its superiority in maximizing QoE is undeniable when compared against all other algorithms. The only other approach that performs well, as C_j increases beyond 90, is QoEUA, but still clearly below FQEA-H. Among the remainder algorithms, DGMIN is the best, with MCF, NS and RND following even further behind and maxing out at around $\mathcal{E}^{\mathcal{A}} = 3.1$, which is far less than FQEA-H, whose QoE reaches $\mathcal{E}^{\mathcal{A}} = 4.3$ under the same conditions.

6.3.3 *Varying number of users*

Fig. 6 captures the results acquired by Scenario #3. Here, we vary the number of potential users. Interestingly, for $|U| \leq 900$, FQEA-H is the only algorithm that assigns 100% of the users to MEC edge servers, as observed in Fig. 6(a). Increasing the users' number negatively affects the admission capabilities of the algorithms, which is ex-

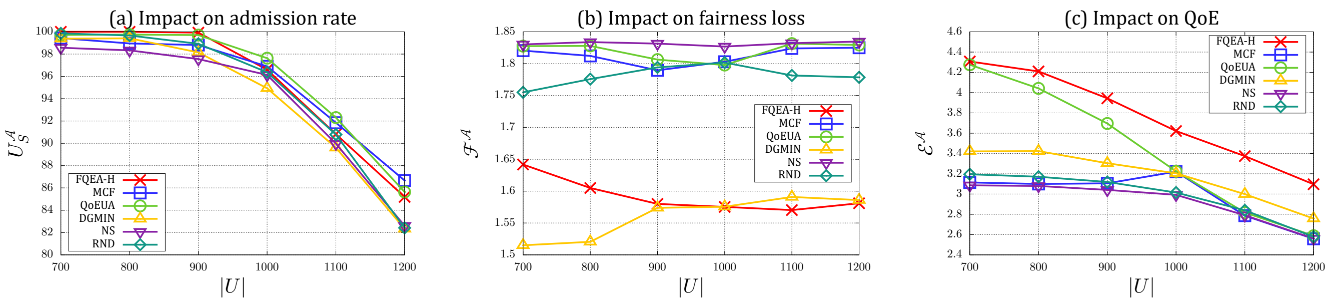


Fig. 6 – (a) User admission, (b) fairness loss, and (c) average QoE as a function of the users' number.

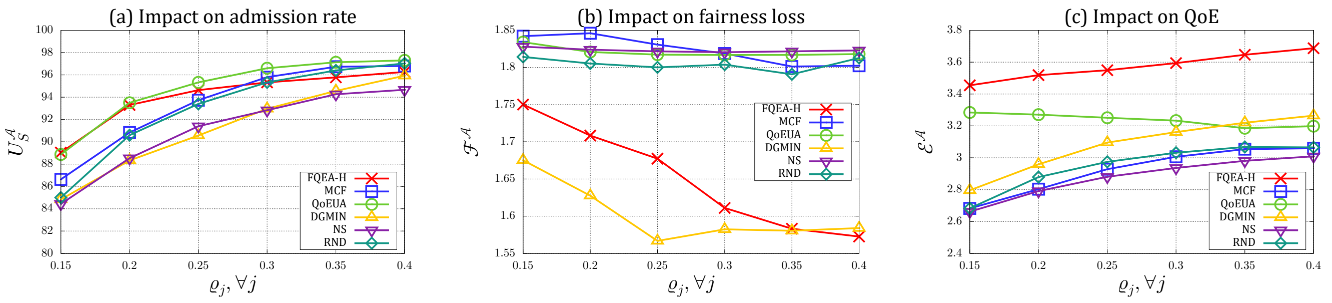


Fig. 7 – (a) User admission, (b) fairness loss, and (c) average QoE as a function of the BSs' coverage range.

pected given the limited capacity and the fixed budget. Therefore, for $|U| \geq 1000$ all approaches experience a drastic decrease in admission rates. Yet, for the most part, our algorithm remains better than most alternatives, with MCF and QoEUA being once more the better approaches in this aspect. However, we must recall that the former is purely designed for user maximization purposes, while the latter employs costly user reallocation during its assignment. That being said, FQEA-H closely approximates their results, different from the rest of the algorithms that experience a more rapid decay in $U_S^{\mathcal{A}}$. Regarding the fairness loss for the VR game groups, it is revealed by Fig. 6(b) that DGMIN and our algorithm again outperform the other alternatives by a significant factor, with the former having the tendency to increase the $\mathcal{F}^{\mathcal{A}}$ as the $|U|$ rises and the latter having the tendency to decrease it even further, giving it the upper hand for $|U| \geq 1000$. The rest of the algorithms perform rather poorly, independently of the users present, a fact that indicates again their unsuitability is optimizing view consistency among the users of the VR games. Note that NS remains the worst algorithm in all cases. As for QoE, it is showcased in Fig. 6(c) that FQEA-H consistently yields the highest $\mathcal{E}^{\mathcal{A}}$ in all circumstances. Actually, although QoEUA starts with an almost similar value in QoE, it is noteworthy that the gulf between the two approaches yawns wider with each successive rise in the $|U|$ value at an accelerated rate, reaching a performance gap of over 17% when $|U| \geq 1100$. Generally, increasing $|U|$ hurts the average QoE for all algorithms, which is justifiable if we consider that the ratio of the users' resource demands to the edge servers' computing capacity also grows with each increment. Still, it is clear that all benchmarks repeatedly report a markedly lesser QoE than FQEA-H under all circumstances. These findings, in conjunction with the decreasing fairness loss and the relatively high admission rate, make FQEA-H the best solution for balancing the trade-off between the multiplayer VR group lag variance and each user's individual QoE, in terms of edge provisioning.

6.3.4 *Varying coverage range*

Fig. 7 illustrates results with respect to Scenario #4, where we vary the coverage range of the BSs hosting the edge servers. Interestingly enough, we can see from Fig. 7(a) that, relating to edge assignment, FQEA-H and QoEUA start with the highest admission rates, performing almost identically, with MCF coming third. For $\varrho_j \geq 0.3$, however, MCF and even RND surpass slightly our algorithm. However, keep in mind that the former is solely aimed at users' edge admission maximization, whereas the latter performs a random but brute search for each user, until its assignment is accomplished. QoEUA, on the other hand, as already mentioned, utilizes greedy reallocation, which in a real-world setting is costly. Even so, the difference in $U_S^{\mathcal{A}}$ among the four algorithms in these cases is negligible and less than 1.4%, making FQEA-H scalable to demanding VR applications. NS and DGMIN also perform adequately but clearly worse under almost all cases. Though FQEA-H reports high admission rate, it nevertheless manages to retain the fairness loss at acceptable levels, as demonstrated in Fig. 7(b). The same cannot be said for the other algorithms, whose curves remain rather stable irrespectively of the coverage range, with the exception of DGMIN whose exclusive goal is the minimization

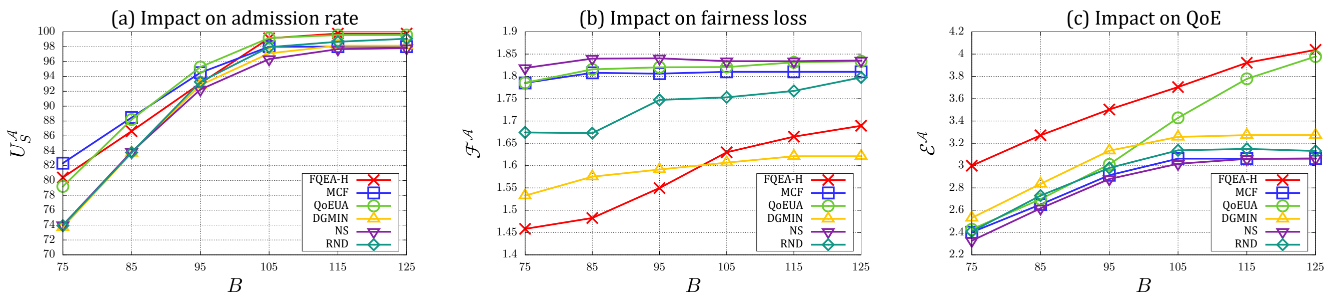


Fig. 8 – (a) User admission, (b) fairness loss, and (c) average QoE as a function of the SP's budget.

of interaction delays. In fact, as ϱ_j expands for all BSs, our algorithm is capable of increasingly reduced $\mathcal{F}^{\mathcal{A}}$, with a fast drop rate. As such, for $\varrho_j \simeq 0.3$ it begins to outperform DGMIN. This attitude is reasonable, because for larger ranges more edge servers become accessible and, thus, FQEA-H can discover increasingly more favorable locations for optimizing the lag variance among the users of each VR game group. All other benchmarks exhibit a somewhat stable performance, that is significantly worse across all cases, with RND, due to its sheer randomness, being marginally better. Evidently, the fairness loss reduction by FQEA-H does not seem to oppose its QoE maximization target, as witnessed in Fig. 7(c). On the contrary, the proposed algorithm once more is superior to all alternatives, with vast QoE gains. For most algorithms, the increase in coverage range benefits $\mathcal{E}^{\mathcal{A}}$, mirroring their increase in $U_S^{\mathcal{A}}$. QoEUA is the only benchmark that seems to be negatively impacted by this rise, which could be attributed to its reallocation rounds that consider QoV but do account for the lag variance caused by increased interserver delays.

6.3.5 *Varying budget*

Finally, Fig. 8 graphically presents results from Scenario #5, where we vary the budget threshold. Assignment-wise, a positive relation between budget raise and admission rate augmentation is reported by all approaches in Fig. 8(a). Nonetheless, we verify that FQEA-H is capable of exceedingly high edge admission rates that are second only to MCF and QoEUA for $B \leq 95$. From that point onward, the proposed algorithm manages to outperform all benchmarks, making the most out of the budget constraint. Ergo, for $B \geq 105$, FQEA-H along with QoEUA yield a $U_S^{\mathcal{A}} \simeq 100\%$, while MCF, DGMIN and NS max out at around 98% and RND at 99%. Meanwhile, on the fairness front, we observe in Fig. 8(b) that FQEA-H initially offers the least $\mathcal{F}^{\mathcal{A}}$, meaning that it strategically opens edge servers given the limited budget. Thus, it prudently outperforms all the alternatives, with NS being the least effective and DGMIN being the closest in comparison. Still, it must be noted that the fairness loss for the proposed algorithm experiences some growth as the budget increases, with a spike for $85 \leq B \leq 105$, which is explainable when coupled with the sharp rise in user assignments during those cases, wherein FQEA-H prioritizes the maximization of $U_S^{\mathcal{A}}$. From thereon, the growth follows a steadily decreasing rate, which indicates that enough servers are now allowed to open in order for FQEA-H to begin again effectively optimizing the maximum lag variance for the VR game groups. Despite this rise in fairness loss by FQEA-H, DGMIN remains the only other algorithm to perform better in this regard. Besides, this pattern must be jointly weighted along with our actual objective, i.e., the QoE maximization, for which FQEA-H is superior under all values of B, as underscored in Fig. 8(C). In fact, it systematically prevails over all benchmarks, with QoEUA being the only alternative capable of yielding comparable $\mathcal{E}^{\mathcal{A}}$ for $B \geq 115$. In contrast, the remainder algorithms begin to converge for $B \simeq 105$. As such, for $B = 125$, which translates to all servers being eligible for user assignment, FQEA-H presents a performance gap of 23.35%, 28.93%, 31.75%, and 31.84% from DGMIN, RND, NS, and MCF respectively. Summarizing, we can remark that FQEA-H is the most suitable solution in tackling FQEA on all fronts, and generally generating elastic assignment strategies that benefit the users in terms of total edge admission, fairness loss, and edge resource provisioning, in order to maximize their QoE and provide them with enhanced group gaming engagement and the best overall VR immersion.

7. CONCLUSION

In this article, we investigated the FQEA problem in VR applications under the edge computing paradigm, where users engaged in various games, given their minimum set of resource requirements in terms of QoV, should witness simultaneously the same scene updates in order to optimize view consistencies and QoS, factors crucial for maximizing their perceived QoE. We theoretically modeled the problem, acknowledging both the limited budget that service providers typically possess for edge server deployment and the idiosyncrasies of the MEC ecosystem itself. We showed that FQEA is a hard problem and subsequently proposed and analyzed FQEA-H, a heuristic algorithm that assigns incoming users strategically, reducing the maximum lag variance experienced by those immersed in the same VR game and in parallel improving their individual QoE. Trace-driven simulation results, against several alternative allocation approaches, testified to FQEA-H's superior performance in finding efficient trade-offs on all fronts, achieving high user edge admission rates, a significant reduction in fairness loss, and the highest overall QoE, making it a suitable solution to future-generation social VR.

There are several avenues for future research. Although we formulated the FQEA under various edge computing parameters, we restricted our analysis to quasi-static topologies as a stepping stone for more complex networks. However, to realistically capture the needs of real-world settings, user mobility must also be considered to address the dynamicity of the MEC environments. Furthermore, QoE in multiplayer VR games depends on many other factors, both objective and subjective, that add to the overall complexity of the problem and should be included in our model. We leave the detailed analysis of these issues to future work.

ACKNOWLEDGEMENT

This work was supported by the project "Corfu Virtual Exhibition Site for Tourism-Culture-Environment" (MIS 5031252), which is partially funded by European and Greek National Funds (ESPA) under the Regional Operational Programme "Ionian Islands 2014-2020".

REFERENCES

[1] Christopher Ball, Kuo-Ting Huang, and Jess Francis. "Virtual reality adoption during the COVID-19 pandemic: A uses and gratifications perspective". In: *Telematics and Informatics* 65 (2021), p. 101728. ISSN: 0736-5853. DOI: https://doi.org/10.1016/j.tele.2021.101728.

[2] Yuan Zhang, Lei Jiao, Jinyao Yan, and Xiaojun Lin. "Dynamic service placement for virtual reality group gaming on mobile edge cloudlets". In: *IEEE Journal on Selected Areas in Communications* 37.8 (2019), pp. 1881–1897. DOI: 10.1109/JSAC.2019.2927071.

[3] Yushan Siriwardhana, Pawani Porambage, Madhusanka Liyanage, and Mika Ylianttila. "A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects". In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 1160–1192. DOI: 10.1109/COMST.2021.3061981.

[4] Sang-Min Park and Young-Gab Kim. "A Metaverse: Taxonomy, Components, Applications, and Open Challenges". In: *IEEE Access* 10 (2022), pp. 4209–4251. DOI: 10.1109/ACCESS.2021.3140175.

[5] Wei Yang Bryan Lim, Zehui Xiong, Dusit Niyato, Xianbin Cao, Chunyan Miao, Sumei Sun, and Qiang Yang. "Realizing the Metaverse with Edge Intelligence: A Match Made in Heaven". In: *arXiv preprint arXiv:2201.01634* (2022).

[6] Zeqi Lai, Y. Charlie Hu, Yong Cui, Linhui Sun, Ningwei Dai, and Hung-Sheng Lee. "Furion: Engineering High-Quality Immersive Virtual Reality on Today's Mobile Devices". In: *IEEE Transactions on Mobile Computing* 19.7 (2020), pp. 1586–1602. DOI: 10.1109/TMC.2019.2913364.

[7] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B. Letaief. "A survey on mobile edge computing: The communication perspective". In: *IEEE Communications Surveys & Tutorials* 19.4 (2017), pp. 2322–2358. DOI: 10.1109/COMST.2017.2745201.

[8] Xiantao Jiang, F. Richard Yu, Tian Song, and Victor C. M. Leung. "A Survey on Multi-Access Edge Computing Applied to Video Streaming: Some Research Issues and Challenges". In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 871–903. DOI: 10.1109/COMST.2021.3065237.

[9] Yongqiang Gao, Chaoyu Zhang, Zhulong Xie, Zhengwei Qi, and Jiantao Zhou. "Cost-Efficient and Quality of Experience-aware Player Request Scheduling and Rendering Server Allocation for Edge Computing Assisted Multiplayer Cloud Gaming". In: *IEEE Internet of Things Journal* (2021). DOI: 10.1109/JIOT.2021.3132849.

[10] Lei Jiao, Jun Lit, Wei Du, and Xiaoming Fu. "Multi-objective data placement for multi-cloud socially aware services". In: *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE. 2014, pp. 28–36. DOI: 10.1109/INFOCOM.2014.6847921.

[11] Haiyang Hu, Rynson WH Lau, Hua Hu, and Benjamin Wah. "On view consistency in multi-server distributed virtual environments". In: *IEEE Transactions on Visualization and Computer Graphics* 20.10 (2014), pp. 1428–1440. DOI: 10.1109/TVCG.2013.244.

[12] Yu Liang, Jidong Ge, Sheng Zhang, Jie Wu, Lingwei Pan, Tengfei Zhang, and Bin Luo. "Interaction-oriented service entity placement in edge computing". In: *IEEE Transactions on Mobile Computing* 20.3 (2021), pp. 1064–1075. DOI: 10.1109/TMC.2019.2952097.

[13] Markus Viljanen, Antti Airola, Jukka Heikkonen, and Tapio Pahikkala. "Playtime Measurement With Survival Analysis". In: *IEEE Transactions on Games* 10.2 (2018), pp. 128–138. DOI: 10.1109/TCIAIG.2017.2727642.

[14] Lu Zhao, Wenan Tan, Bo Li, Qiang He, Li Huang, Yong Sun, Lida Xu, and Yun Yang. "Joint Shareability and Interference for Multiple Edge Application Deployment in Mobile Edge Computing Environment". In: *IEEE Internet of Things Journal* 9.3 (2022), pp. 1762–1774. DOI: 10.1109/JIOT.2021.3088493.

[15] Lin Wang, Lei Jiao, Ting He, Jun Li, and Max Mühlhäuser. "Service entity placement for social virtual reality applications in edge computing". In: *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE. 2018, pp. 468–476. DOI: 10.1109/INFOCOM.2018.8486411.

[16] Phu Lai, Qiang He, John Grundy, Feifei Chen, Mohamed Abdelrazek, John G Hosking, and Yun Yang. "Cost-effective app user allocation in an edge computing environment". In: *IEEE Transactions on Cloud Computing* (2020). DOI: 10.1109/TCC.2020.3001570.

[17] Chunrong Wu, Qinglan Peng, Yunni Xia, Yong Ma, Wangbo Zheng, Hong Xie, Shanchen Pang, Fan Li, Xiaodong Fu, Xiaobo Li, et al. "Online user allocation in mobile edge computing environments: A decentralized reactive approach". In: *Journal of Systems Architecture* 113 (2021), p. 101904. DOI: 10.1016/j.sysarc.2020.101904.

[18] Guangming Cui, Qiang He, Feifei Chen, Hai Jin, and Yun Yang. "Trading off between user coverage and network robustness for edge server placement". In: *IEEE Transactions on Cloud Computing* (2020). DOI: 10.1109/TCC.2020.3008440.

[19] Tingting Li, Wenqi Niu, and Cun Ji. "Edge user allocation by FOA in edge computing environment". In: *Journal of Computational Science* 53 (2021), p. 101390. DOI: https://doi.org/10.1016/j.jocs.2021.101390.

[20] Xiao Ma, Shangguang Wang, Shan Zhang, Peng Yang, Chuang Lin, and Xuemin Shen. "Cost-Efficient Resource Provisioning for Dynamic Requests in Cloud Assisted Mobile Edge Computing". In: *IEEE Transactions on Cloud Computing* 9.3 (2021), pp. 968–980. DOI: 10.1109/TCC.2019.2903240.

[21] Athanasios Tsipis and Konstantinos Oikonomou. "ARPA: An autonomous renderer placement algorithm in distributed multimedia fog networks with delay guarantees". In: *ITU Journal on Future and Evolving Technologies* 2.2 (2021), pp. 81–108. DOI: 10.52953/TSDT6782.

[22] Phu Lai, Qiang He, Guangming Cui, Xiaoyu Xia, Mohamed Abdelrazek, Feifei Chen, John Hosking, John Grundy, and Yun Yang. "QoE-aware user allocation in edge computing systems with dynamic QoS". In: *Future Generation Computer Systems* 112 (2020), pp. 684–694. DOI: https://doi.org/10.1016/j.future.2020.06.029.

[23] Songyuan Li, Jiwei Huang, Jia Hu, and Bo Cheng. "QoE-DEER: A QoE-Aware Decentralized Resource Allocation Scheme for Edge Computing". In: *IEEE Transactions on Cognitive Communications and Networking* (2021), pp. 1–1. DOI: 10.1109/TCCN.2021.3118460.

[24] Hanying Zheng and Xueyan Tang. "The server provisioning problem for continuous distributed interactive applications". In: *IEEE Transactions on Parallel and Distributed Systems* 27.1 (2016), pp. 271–285. DOI: 10.1109/TPDS.2015.2388473.

[25] Haiyang Wang, Tong Li, Ryan Shea, Xiaoqiang Ma, Feng Wang, Jiangchuan Liu, and Ke Xu. "Toward cloud-based distributed interactive applications: measurement, modeling, and analysis". In: *IEEE/ACM Transactions on Networking* 26.1 (2018), pp. 3–16. DOI: 10.1109/TNET.2017.2765246.

[26] Seyhan Ucar, Huseyin Guler, and Oznur Ozkasap. "Online Client Assignment in Dynamic Real-Time Distributed Interactive Applications". In: *2013 IEEE/ACM 17th International Symposium on Distributed Simulation and Real Time Applications*. IEEE. 2013, pp. 65–71. DOI: 10.1109/DS-RT.2013.15.

[27] Haoyu Zhu, Yingjiao Li, Zhiyong Chen, and Li Song. "Mobile Edge Resource optimization for Multiplayer Interactive Virtual Reality Game". In: *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE. 2021, pp. 1–6. DOI: 10.1109/WCNC49053.2021.9417124.

[28] Alaa Eddin Alchalabi, Shervin Shirmohammadi, Shady Mohammed, Sorin Stoian, and Karthigesu Vijayasuganthan. "Fair Server Selection in Edge Computing With Q-Value-Normalized Action-Suppressed Quadruple Q-Learning". In: *IEEE Transactions on Artificial Intelligence* 2.6 (2021), pp. 519–527. DOI: 10.1109/TAI.2021.3105087.

[29] Paniz Parastar, Farhad Pakdaman, and Mahmoud Reza Hashemi. "FRAME-SDN: a fair resource allocation for multiplayer edge-based cloud gaming in SDN". In: *Proceedings of the 25th ACM Workshop on Packet Video*. 2020, pp. 21–27. DOI: 10.1145/3386292.3397120.

[30] Ran Xu and Weiqiang Zhang. "Improving Fairness for Distributed Interactive Applications in Software-Defined Networks". In: *Mathematical Problems in Engineering* 2020 (2020). DOI: 10.1155/2020/5207105.

[31] Athanasios Tsipis and Konstantinos Oikonomou. "Player Assignment in MEC Gaming for Social Interactivity and Server Provisioning Optimization". In: *2021 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2021, pp. 1–7. DOI: 10.1109/ISCC53001.2021.9631480.

[32] Athanasios Tsipis and Konstantinos Oikonomou. "Joint optimization of social interactivity and server provisioning for interactive games in edge computing". In: *Computer Networks* 212 (2022), p. 109028. DOI: https://doi.org/10.1016/j.comnet.2022.109028.

[33] Vajiheh Farhadi, Fidan Mehmeti, Ting He, Thomas F La Porta, Hana Khamfroush, Shiqiang Wang, Kevin S Chan, and Konstantinos Poularakis. "Service placement and request scheduling for data-intensive applications in edge clouds". In: *IEEE/ACM Transactions on Networking* 29.2 (2021), pp. 779–792. DOI: 10.1109/TNET.2020.3048613.

[34] Kuan-Ta Chen, Yu-Chun Chang, Hwai-Jung Hsu, De-Yu Chen, Chun-Ying Huang, and Cheng-Hsin Hsu. "On the Quality of Service of Cloud Gaming Systems". In: *IEEE Transactions on Multimedia* 16.2 (2014), pp. 480–495. DOI: 10.1109/TMM.2013.2291532.

[35] Maciej D. Sobociński. "Quality of video games: introduction to a complex issue". In: *Quality Production Improvement - QPI* 1.1 (2019), pp. 487–494. DOI: 10.2478/cqpi-2019-0066.

[36] Feifei Chen, Jingwen Zhou, Xiaoyu Xia, Hai Jin, and Qiang He. "Optimal application deployment in mobile edge computing environment". In: *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*. IEEE. 2020, pp. 184–192. DOI: 10.1109/CLOUD49709.2020.00037.

[37] Athanasios Tsipis and Vasileios Komianos. "Scalable Server Location for Distributed Interactive Applications Under Budget Constraints". In: *2021 International Balkan Conference on Communications and Networking (BalkanCom)*. 2021, pp. 21–25. DOI: 10.1109/BalkanCom53780.2021.9593261.

[38] Lu Zhang and Xueyan Tang. "The client assignment problem for continuous distributed interactive applications: Analysis, algorithms, and evaluation". In: *IEEE Transactions on Parallel and Distributed Systems* 25.3 (2014), pp. 785–795. DOI: 10.1109/TPDS.2013.47.

[39] Kuan-Ta Chen, Polly Huang, and Chin-Laung Lei. "Effect of Network Quality on Player Departure Behavior in Online Games". In: *IEEE Transactions on Parallel and Distributed Systems* 20.5 (2009), pp. 593–606. DOI: 10.1109/TPDS.2008.148.

[40] Chengchao Liang, Ying He, F. Richard Yu, and Nan Zhao. "Enhancing Video Rate Adaptation With Mobile Edge Computing and Caching in Software-Defined Mobile Networks". In: *IEEE Transactions on Wireless Communications* 17.10 (2018), pp. 7013–7026. DOI: 10.1109/TWC.2018.2865354.

[41] Hua-Jun Hong, De-Yu Chen, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. "Placing Virtual Machines to Optimize Cloud Gaming Experience". In: *IEEE Transactions on Cloud Computing* 3.1 (2015), pp. 42–53. DOI: 10.1109/TCC.2014.2338295.

[42] Yiwen Han, Dongyu Guo, Wei Cai, Xiaofei Wang, and Victor Leung. "Virtual Machine Placement Optimization in Mobile Cloud Gaming through QoE-Oriented Resource Competition". In: *IEEE Transactions on Cloud Computing* (2020), pp. 1–1. DOI: 10.1109/TCC.2020.3002023.

[43] Mahdi Hemmati, Bill McCormick, and Shervin Shirmohammadi. "QoE-Aware Bandwidth Allocation for Video Traffic Using Sigmoidal Programming". In: *IEEE MultiMedia* 24.4 (2017), pp. 80–90. DOI: 10.1109/MMUL.2017.4031305.

[44] Silvano Martello and Paolo Toth. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.

[45] Phu Lai, Qiang He, Mohamed Abdelrazek, Feifei Chen, John Hosking, John Grundy, and Yun Yang. "Optimal edge user allocation in edge computing with variable sized vector bin packing". In: *International Conference on Service-Oriented Computing*. Springer. 2018, pp. 230–245. DOI: 10.1007/978-3-030-03596-9_15.

AUTHORS

Athanasios Tsipis received in 2015 his B.Sc. in Informatics from the Department of Informatics, Ionian University, Greece. In 2017 he obtained his M.Sc. in Informatics, for which he was awarded a scholarship for academic excellence, while in 2021 he received his Ph.D. in Informatics with honors, from the same Department. His research interests concentrate on networked immersive systems, immersive technologies, multimedia cloud computing, cloud/edge gaming, network optimization and performance issues, service placement, location-allocation problems, metaverse applications, interaction design, digital culture, etc. Since 2015 he has been an active member and researcher of the "Networks, Multimedia and Security Systems Laboratory" (NMSLab), participating in various EC research and local development projects. In 2021, he was with the Department of Informatics at Ionian University, as an academic scholar. Currently, he serves as an adjunct lecturer at the Department of Digital Media and Communication, as well as an academic scholar at the Department of Tourism, at Ionian University. He has been a reviewer of numerous conferences and journals, and in 2021 he was the recipient of the best paper award from the 26th IEEE Symposium on Computers and Communications.

Vasileios Komianos is an Assistant Professor in Mixed Reality Systems at the Department of Audio and Visual Arts, Ionian University, Greece, teaching courses related to virtual/augmented/mixed reality, video games and interactive multimedia. His research interests are mostly focused on Mixed Reality (MR) systems, on user interaction and user interfaces in MR systems and applications as well as on approaches for artistic expression and cultural communication. He has work experience in designing audiovisual content and installations in the cultural heritage sector, and his work is hosted or has been hosted in permanent and temporary exhibitions as well as in art festivals.

Konstantinos Oikonomou received his MEng in Computer Engineering and Informatics from the University of Patras in 1998. In September 1999 he received his M.Sc. degree in Communication and Signal Processing from the Electrical and Electronic Engineering Department, Imperial College (London). He received his Ph.D. degree in 2004 from the Informatics and Telecommunications Department, University of Athens, Greece. His Ph.D. thesis focuses on medium access control policies in ad hoc networks. He has served as the Dean of the Faculty of Information Science and Informatics of the Ionian University, Corfu, Greece. Since 2006 he is a faculty member in Computer Networks, currently a Professor, at the Department of Informatics, at the same University. He has also served as the Head of the Department and Director of the Postgraduate Studies Program. Between December 1999 and January 2005, he was employed at Intracom S.A, as a research and development engineer. His research interests involve medium access control in ad hoc networks, performance issues in wireless networks, information dissemination, service discovery, facility location, energy consumption in wireless sensor networks, and network cost reduction in cloud computing environments. Professor K. Oikonomou has extensive experience in wireless systems and has been coordinating a number of EC research projects in the computer networks area, and various local development projects (e.g., virtual worlds). He is currently a member of the editorial boards of Computer Networks Journal (Elsevier) and Journal on Future and Evolving Technologies (ITU). He has been a reviewer and TPC member of numerous conferences and journals, and he holds an award for the best paper from the Hawaii International Conference on System Service, as well as an award for the best paper from the 26th IEEE Symposium on Computers and Communications.

Ioannis Stavrakakis (IEEE Fellow) received the Ph.D. degree from the University of Virginia, USA, in 1988. He has held faculty positions with Northeastern University (1994–1999) and the University of Vermont (1988–1994), following the completion of his Ph.D. studies. He is currently a Professor with the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Greece, and has served as its chair in 2012–2016. He has recently been a Visiting Professor of the University Carlos III de Madrid (UC3M) and the IMDEA Networks Institute. He was a Visiting Professor of the Politecnico di Torino and a Mercator Fellow of the German Research Foundation-DFG at the MAKI Centre at TU Darmstadt. He has authored over 250 publications, supervised 20 Ph.D. graduates, and has been funded by USA-NSF, DARPA, GTE, BBN and Motorola (USA), Greek, and EU agencies, including two Marie-Curie grants (post-doctoral researchers). He has served for NSF and EU-IST proposal panels and conference organization sponsored by IEEE, ACM, ITC, and IFIP. His research interests in networking: social, mobile, ad-hoc, information-centric, delay tolerant, and future Internet networking; network resource allocation algorithms and protocols, traffic management, and performance evaluation; and (human-driven) decision making in competitive environments. He was a recipient of the Chair of Excellence Comunidad de Madrid and the UC3M/Santander Chair of Excellence. He has served as the Chairman for IFIP WG6.3 and an Officer for IEEE Technical Committee on Computer Communications (TCCC). He has served on the editorial boards for Proceedings of the IEEE, IEEE/ACM TRANSACTIONS ON NETWORKING, and Computer Communications journals.