Liberty: An Economy, Three Fold Collaboration, and Virtualization-aware Resource Management Approach for Multiverse Based Application Execution Over 6G Enhanced Networks

Mahfuzulhoq Chowdhury

***Abstract*—Metaverse-based virtual worlds can provide users with an immersive digital experience by utilizing extended reality, IoT, 6G communication, and computing technology. Unlike the multiverse, in which users can access one virtual environment-based activity at a time, metaverse applications allow users to engage in different virtual environment-based activities simultaneously. Previous research did not look at three-fold collaboration, time-criticality, economy, energy, and network function virtualization (NFV)-aware resource management for multiverse, metaverse, and non-metaverse applications concurrently. To conquer these issues, this article proposes a three-fold collaboration (usage of host, neighbor user, and service provider-owned resources), time-criticality, NFV, energy, economy, and service delivery delay-aware resource management scheme for multiverse, metaverse, and non-metaverse application execution over 6G networks, taking into account heterogeneous physical and digital workers, NFV, communication resources, application arrival, and deadline. Experimental results show that the proposed liberty scheme outperforms existing methods by at least 22.67% economic cost, 31.28% energy expenditure cost, 44.11% service provider profit, and 8.83% task completion ratio gain.**

***Index Terms*—Metaverse, Resource Management, Multiverse, 6G, Application and Worker Scheduling, Mobile-edge Computing (MEC), Network Function Virtualization (NFV), Blockchain.**

I. Introduction

THE metaverse has emerged as a leading contender to replace today's Internet. Rather than using traditional internet facilities (enabled by Web 2.0 technologies), such as navigating webpages with a laptop or smartphone, individuals can communicate in the metaverse through their avatar in 3D virtual environments, as well as work, learn, or spend quality time in a virtual setting. Users in the metaverse can enjoy an immersive virtual experience powered by various tactile and haptic feedback sensors using a variety of virtual visualization technologies such as virtual reality (VR), extended reality (XR), mixed reality (MR), and augmented reality (AR) [1]. Currently, many multibillion-dollar corporations are eager to invest in exciting metaverse-based applications (such as immersive gaming and non-fungible token transfer-based virtual trading applications). Facebook recently rebranded as Meta and plans to invest 10 billion USD in its Metaverse divisions [2]. Microsoft invested USD 70 billion to acquire activision blizzard, a metaverse-based video game company [3]. Metaverse applications, which offer immersive digital experiences through Avatar and XR devices, have grown rapidly in recent years. The consumer technology association estimates that AR/VR-based healthcare applications, such as VR-based disease diagnosis, will generate 7 billion US dollar by 2026 [4]. According to a market report [5], AR/VR-based metaverse applications in education (e.g., video conferencing, remote learning via avatar) can grow up to 18% annually compared to current income over the next five years. To connect users' IoT (Internet of things) devices (e.g., Hololens, XR devices) to the virtual world created by the metaverse, seamless network connectivity with massive user support, very high reliability, mobile edge computing (MEC) based cloud computing, and very low latency communication is required [6], [7].

The term "metaverse" is formed by combining the words "universe" (one world) and the prefix "meta" (beyond). The term metaverse (beyond one world) refers to an imagined, man-made setting (virtual world) that is interconnected with the real physical world, which was first suggested by neal stephenson in his 1992 science fiction book, snow crash [8]. The works in [9], [10], [11], and [12] elaborated that the term "metaverse" refers to a sophisticated virtual environment that combines digital and physical world elements by coordination and use of 5G/6G network technologies, web technologies, cloud technologies, blockchain (e.g., cryptocurrency exchange, non-fungible token (NFT) based trading, metaverse coin exchange platform like sandbox, decentraland), digital twin (e.g., healthcare simulation work), artificial intelligence (for intelligent and sophisticated metaverse platform), IoT, and extended reality (XR) technologies.

The multiverse is another popular term used to describe the metaverse. There are some differences between the multiverse and the metaverse. The multiverse (e.g., multiple eco-systems of different universes or metaverses/virtual platforms) offers access to virtual environment-based applications via XR technologies, where users can engage in different activities, such as enjoying music, walking, and video game playing [13], [14]. However, their ability to complete only one task at a time distinguishes them fundamentally. Every project or work in the metaverse (i.e., shared virtual world, all activities within the same ecosystem), including games, trading, tours, and museums, may be housed on a single platform (e.g., one planet).

In other words, in the metaverse (where applications or tasks are linked to a single universe or platform), a user can

Manuscript received April 24, 2024; revised September 8, 2024.

Mahfuzulhoq Chowdhury is a professor at the Computer Science and Engineering Department, Chittagong University of Engineering and Technology, Chittagong-4349, Bangladesh (e-mail: mahfuz@cuet.ac.bd).

enter both a game and an art-selling market. In a multiverse, the host user can play the video game but cannot also access an NFT-based trading platform for virtual art sales. To summarize, a multiverse is a collection of distinct virtual realities that you can switch between to accomplish various tasks [15].

The metaverse is expected to significantly impact every aspect of human life (e.g., from entertainment to business to education to transportation) through its promising applications such as virtual workplaces, holographic telepresence-based remote video conferencing, digital twin-based simulations, virtual trading via non-fungible token transfer and blockchain-based security, immersive gaming, virtual tours or concert experiences, virtual museum tours, and 360-degree However, to provide users with real-time quality of service (QoS), metaverse applications must meet specific requirements, such as ultra-low latency, high resource demands, coordination of physical and virtual world resources, application compatibility, security enhancements, and privacy enhancements [16], [17].

Currently, central cloud-based solutions are used for some metaverse-based application execution tasks such as avatar creation and emulation, rendering graphics, adding static data to video, 3D animation, and creating new environments, among others. Such centralized cloud-based implementations are undesirable for a variety of reasons, including poor visualization quality, increased collisions, very long queuing delays, and extremely high cloud access wide-area network communication latency. To improve the existing central cloud-based server processing issues, a separate region-based cloud computing or edge cloud server (mobile-edge cloud computing, or MEC) would be advantageous for avatar-based metaverse application execution due to its low computational and communication latency [18]. Using XR technology, the metaverse can create an immersive virtual environment that interacts with both physical and virtual worlds. The metaverse allows users to work, talk, communicate, and interact with other users and the real world through avatars. Metaverse-based application processing (e.g., creating avatars, simulated or virtual environments) requires significant computational and storage resources [19], [20]. User devices (e.g., smart phones, robots) may not be able to process computation and data-intensive (latency-restricted work) metaverse services alone due to insufficient computation and storage resources [21].

To overcome the resource limitation issue of user devices, the computation offloading technique can be used, which allows the user device to offload the full or partial metaverse task workload to the MEC server for processing [20], [21]. The research community faces a challenge in allocating MEC and communication resources, selecting a suitable MEC server, and meeting the QoS requirements of multiple metaverse users [21].

Currently, there are several articles on joint computation task offloading, server placement, and routing issue problem solving for MEC-based networks [22], [23], [24], [25]. However, the majority of the literature articles focus on single computation task offloading issues, single MEC servers, and single types of resource allocation rather than addressing task offloading and resource allocation issues for multiple meatverse-based task offloading problems. Numerous literary works have examined the research challenges and potential technology selection issues for metaverse application implementation [26], [27], [28], [29]. In [27] and [30], the authors discussed the recent developments, enablers, and architectures to realize wireless technology-based metaverse application execution. In [28], the authors provided a detailed summary of several artificial intelligence (AI) approaches along with other suitable technologies like computer vision, blockchain, MEC, and natural language processing (NLP).

Previous studies [29], [30], and [31] addressed the issue of physical and digital worker interactions in the metaverse. However, their work is restricted to physical worker/avatar modeling or avatar creation methods, rather than resource allocation and appropriate physical and digital worker selection for metaverse and multiverse-based application. In [32], researchers used federated learning and blockchain to execute data-sharing-based metaverse applications, ensuring user data security and privacy. Thus, proper coordination between computation networks and data-intensive networks, as well as proper resource management during application work processing, are required for emerging metaverse applications. In [33], the authors explored blockchain's potential for metaverse applications, including secure data acquisition, storage, interoperability, and privacy. In [34], researchers investigated how blockchain, AI, and edge computing can collaborate to enable seamless access to metaverse applications. The article [35] presents a detailed metaverse taxonomy. The authors categorize metaverse technologies based on different hardware, software, applications, user interaction types, and implementation processes after examining three use cases (for example, the ready player one event, the roblox gaming platform). In [19], the authors discussed different steps associated with metaverse application development, such as physical world and virtual world integration, digital twin creation for both humans and IoT devices, initial application input or content creation, interoperability, MEC-based content processing, and visualization. The authors of [36] conducted a survey on metaverse fundamentals, security challenges, and privacy solutions. The article in [37] discussed several promising technologies for developing green metaverse applications that save energy. They discussed 6G communication and computation technologies, IoT, digital twin (DT), AI, MEC, blockchain, and various reality-based technologies for metaverse. In [38], researchers investigated various metaverse issues, including digital content generation, decentralization, interoperability issues, computation, and storage issues. They also presented a four-layer MEC-enabled network architecture for metaverse.

However, the aforementioned study did not investigate economic and energy cost-conscious resource management and worker selection for various metaverse application executions. Articles in [39] and [40] suggest that 6G technology and edge AI (artificial intelligence technology) can effectively connect virtual and physical worlds in the metaverse. Many metaverse applications can benefit from 6G technologies, which include very low latency, massive connectivity, energy efficiency, and seamless and ubiquitous connectivity.

Currently, 6G communication and computation technologies (e.g., terahertz wave, optical-wireless link connectivity, MEC, machine and deep learning, blockchain, AI, digital twin, XR technologies) are regarded as enabling technologies for different metaverse applications. They can support data rates greater than 1 Tbps, network reliability and availability up to 99.99999%, mobility support near 1000 Kmph, high connection density over 100 devices/km^2, edge processing delays lower than 10 ns, and end-to-end delay lower than .1 ms [40] and [41]. The development of 6G applications is expected to focus on new requirements such as massive, ultra-reliable, and low-latency communication (mULC) for human-machine coworking, ubiquitous mobile broadband (uMBB), digital twin-based virtual healthcare services, autonomous driving applications, and ultra-reliable and low-latency broadband (ULBC) services. According to [42] and [43], 6G application requirements can be met by integrating and coordinating various 6G technologies, such as THz and mmwave spectrum usage, machine and deep learning technology, edge AI technology, blockchain, MEC, and RIS (reconfigurable interference).

According to [43], network slicing is an important issue for 6G metaverse apps. The authors of [9] discussed the feasibility of implementing the metaverse on mobile edge networks, including blockchain operations, computing tasks, communication facilities, and networking. According to [44] and [45], a network slice can be allocated for 6G application execution (e.g., metaverse-based virtual gaming, industrial manufacturing) by taking both physical and digital device resources, as well as communication, storage, and virtual network resources. Another critical issue is the order of virtual network processing (VNF), server selection for 6G metaverse application execution, and proper resource slicing. Traditionally, network functions (such as firewalls and load balancers) were implemented on dedicated hardware appliances known as middleboxes. The cost of designing and manufacturing hardware middleboxes (i.e., dedicated hardware for network service provisioning, such as firewalls, intrusion detection, load balancing, and digital packet inspection) makes their implementation expensive. Furthermore, the manual configuration and management of these middleboxes raises service provider costs [46]. To address these issues, network function virtualization (NFV) technology is a popular solution as it allows software instances to run in a virtual environment, replacing hardware-based implementations [47]. NFV delivers the required service via a set of virtual network functions (VNFs). Virtualization technology enables these virtual network functions (VNFs) to run on standard servers. Service Function Chaining (SFC) is an organized collection of VNFs that manage traffic for a specific application (i.e., an ordered set of VNF execution) [48]. NFV provides a more effective and flexible framework for managing and operating network services, allowing for more scalable and elastic resource allocation, resulting in a significant reduction in overall costs. To reduce end-to-end delay in SFC-based VNF execution over the NFV-enabled network, proper placement and selection of the VNF server is necessary, considering the delay-sensitive nature of the applications [49]. It should be noted that violations of SLAs (service level agreements, such as deadline satisfaction) during VNF execution may reduce the service provider's overall profit. As a result, proper VNF packet routing, orderly VNF execution, and efficient server selection for VNF processing are required.

There are some research articles about VNF execution. In [50], a VNF routing and placement problem using integer linear programming (ILP) was developed to reduce operational costs. They also used heuristics to solve the ILP-based optimization problem. The article in [51] proposed a traffic load-aware VNF server selection scheme that can maximize link load and reduce the specific VNF server dependency problem. The study in [52] used a VNF placement scheme based on Markov approximation to reduce operational and traffic costs. In [53], a heuristic solution was provided to a min-cost flow problem for VNF placement. The authors of [54] proposed a hungarian-based resource allocation technique for blockchain-enabled systems that reduces energy usage and costs. According to [55], the main challenge for SFC-based VNF execution is to select a suitable VNF server and allocate computation and communication resources for task execution.Another critical issue is to reduce the end-to-end delay for SFC-based VNF processing task execution. The work in [56] focused on minimizing server access waiting delays for VNF processing, rather than minimizing transmission, waiting, and computation delays concurrently. The articles in [57] and [58] studied the problem of minimizing transmission delay during SFC-based multiple VNF processing task execution.

The article in [59] discussed research issues and challenges in SFC-based VNF execution, such as VNF server management, VNF execution order selection, routing policy, and resource management, among others. The article in [58] proposed a BFS-based SFC deployment optimization policy. They also presented results comparing the proposed scheme to the existing scheme (for example, greedy versus simulated annealing-based SFC deployment). The article in [60] provided a systematic review of the use of NFV in data center networks. The authors of [61] created a constrained combinatorial optimization scheme with a deep reinforcement learning-based solution to minimize the cost of VNF reconfiguration. In [62], the authors proposed a resource ability-based VNF scheduling model with FCFS (first come, first served). The authors of [63] proposed an ILP-based optimization model for application-aware VNF mapping in heterogeneous NFV environments. According to [64], NFV-enabled metaverse application execution requires a service level agreement-aware resource allocation policy that considers various latencies, including computing latency (digital and physical worker task execution latency, as well as network function processing latency), communication latencies, and waiting latencies. In [65], researchers proposed a two-stage optimization problem for selecting resources and services in a virtual network. In [66], the authors discussed the necessity and integration of blockchain, edge computing, and software-defined network technology to achieve wireless network virtualization. The article in [67] examined load-balancing-based VNF server selection in wireless networks. Previous research [65], [66], [67] focused on a single type of resource allocation (e.g., VNF resource) rather than multiple types (e.g., computing, storage, communication resource). Furthermore, their analysis did not include an examination of different metaverse application types, worker selection, physical and digital world integration, or SFC-based VNF provisioning for metaverse application execution. The work in [68] emphasizes the on-demand resource orchestration scheme for metaverse application execution, taking into account different types of resource assignment (e.g., physical/virtual node-based computation and communication resource selection). In [69], the authors formulated an optimization problem based on mixed integer liner programming for joint VNF placement and traffic routing issues while maintaining high user acceptance rate for 6G heterogeneous network-based applications. In [70], a traffic prediction algorithm based on q-learning is proposed to embed VNFs in fiber-wireless networks with resource efficiency.

According to the previous literary work discussion, previous studies did not provide an appropriate resource management scheme for the execution of various multiverse, metaverse, and non-metaverse applications over 6G-enabled MEC networks. They did not investigate any resource selection and application coordination schemes that took three-way collaboration into account, as well as the use of host user-owned resources, neighbor user-owned resources, and service provider-owned resources for various multiverse, metaverse, and non-metaverse application implementations. The existing literary works did not present any economy, energy, and time-criticality-aware worker node assignment scheme for different multiverse and non-metaverse-based 6G applications by taking into account not only physical worker resources (e.g., user device, XR device, cobot, smart phone, vehicle) but also digital worker resources (e.g., MEC server, caching server, federated learning or FL server, digital twin server, and blockchain work processing server) but also both VNF resources and communication link resources. Without an appropriate resource scheduling and worker selection system, 6G-based multiverse, metaverse, and non-metaverse-based application execution services may face significant delays in application service delivery, increased energy expenditure and economic cost for host users, lower throughput, a lower successful task completion ratio, and lower service provider profit.

The existing works did not answer the important research question of how to schedule multiple multiverse, metaverse, and non-metaverse-based 6G applications for users while accounting for various available MEC server and caching server resources, host user-owned, neighbor-owned, and service provider-owned resources, VNF server resources, communication resources, energy and economic budget, application deadline or time criticality requirements, and over 6G-enabled edge networks. Previous work did not explore how to coordinate physical, virtual network processing worker, and digital worker-based multiverse, metaverse, and non-metaverse service delivery over 6G-enabled MEC networks with blockchain, caching, XR services, federated learning, VNF processing, digital twin, and federated learning (FL)-aware activities. The previous research projects failed to provide a suitable 6G network model for economy, energy, time criticality, NFV-aware multiverse, metaverse, and non-metaverse application work completion. Crucially, previous research projects did not provide any performance study results for the execution of heterogeneous multiverse, metaverse, and non-metaverse applications using host users, collaborative neighbor users, and service provider-owned resources (e.g., MEC server, cache server, NFV processing server, blockchain node, FL processing node, user device, XR device, and digital twin server). Most importantly, previous literature failed to provide an appropriate numerical model for analyzing the performance of multiple multiverse, metaverse, and non-metaverse-based 6G edge computing networks by considering various factors such as mean application service delivery delay, economic cost, energy expenditure, successful task completion ratio, throughput ratio, service provider total profit, user total benefit ratio or welfare, unused energy, network lifetime, and user. Nonetheless, several delay items, such as elementary network formation phase delay, information collection and resource mapping phase delay, the physical worker-based workload processing delay, the digital worker-based computation workload processing delay (e.g., MEC server, blockchain, FL server processing delay), communication delay, the virtual network function processing delay, and the resource access waiting delay were not factored into the previous works mean application service delivery delay calculation for 6G application execution (e.g., multiverse, metaverse, and non-metaverse). Furthermore, the previous research lacked a comprehensive performance analysis result visualization and comparison with existing schemes for calculating users' energy expenditure costs, economic costs, throughput, task completion ratio, profit value for service providers, network lifetime,user survivability ratio, and total benefit ratio for users.

In order to overcome the current limitations, this article brings a three-fold collaboration among resources, network function virtualization, users economic cost, energy expenditure cost, time-criticality-aware resource management, worker alocation, work coordination, and application scheduling policy for multiverse, metaverse, and non-metaverse-based application execution over 6G edge computing networks by taking different types of application and resource availability into account. In the following, the primary contributions of this article are hinted below:

- This paper proposes a resource management policy that allocates the best physical worker, digital worker, and NFV processing server, as well as the best communication resource allocation, to various multiverse, metaverse, and non-metaverse-based 6G application services. It does this by accounting for various factors such as heterogeneous multiverse, metaverse, and non-metaverse application requests, different deadlines, heterogeneous resource and worker availability, blockchain, digital twin, federated learning, MEC server-based computing and caching operations, physical and digital device-based work processing, collaboration, and utilization of threefold resources (including host user, neighbor-owned resources, and service provider-owned resources), and different economic costs for different services.
- This paper provides a multi-application resource scheduling and physical/digital worker coordination scheme by taking different VNF processing-enabled multiverse, metaverse, and non-metaverse applications into account. The metaverse application request includes XR-based immersive gaming, avatar-based virtual tours of museums, avatar-based virtual clothing experiences, XR-based remote healthcare, socialization through avatar interaction, hologram-based remote presence for video conferencing, federated learning-based surveillance applications, NFT (non-fungible token) and blockchain-enabled visual art selling, and digital twin-based robot selection for manufacturing

jobs. The non-metaverse application includes a human-digital worker-cobot-video sensor-based industrial automation job, autonomous driving application, traditional broadband-based data exchange application, video file caching application, brain sensor-digital worker-wheelchair interaction-based biological application porcessing, and charging station-to-electric vehicle energy transfer application. The multiverse application request includes combined avatar socialization and NFT-based trading work, as well as combined virtual museum tours and virtual clothing experience work via XR and haptic devices.

- This article presents a novel network service model for 6G, MEC,and NFV-enabled metaverse, multiverse, and non-metaverse application execution that includes host users, neighbor users, and service provider-owned digital servers (e.g., MEC servers, remote cloud servers, caching servers, blockchain nodes, VNF servers, digital twin servers, federated learning servers), physical devices (e.g., XR devices, robots, cobots, vehicles, sensors, smart phones, holographic screens, wheelchairs), user access, transport, and core networks, 6G technology-based fiber and wireless communication devices (cellular base stations and WiFi access points), and data transfer services.
- This paper describes a numerical analysis model that provides the mathematics behind the calculation of multiple performance metrics, including mean application service delivery delay, economic cost, energy expenditure, successful task completion ratio, throughput ratio, service provider total profit, user total benefit ratio or welfare, unused user energy, network lifetime value, and user survivability ratio. In contrast to previous research, the mean application service delivery delay includes several delay items such as the elementary network formation phase delay, information collection and resource mapping phase delay, physical worker-based workload processing delay, digital worker-based computation workload processing delay, communication delay, virtual network function processing delay, and resource access waiting time delay.
- This article depicts the simulation results of the proposed liberty scheme with economy, time-criticality, three-fold collboration among resources (with host user, neighbor, and service provider ownership-based resource usage), shortest job first scheduling, and NFV-aware resource management policy. The performance of the proposed liberty scheme is compared with existing scheme one (with no user energy saving policy, double collaboration with user and service provider resource usage, earliest application arrival-based resource scheduling), and existing scheme two (single collaboration with only service provider resource usage, random scheduling-based resources, and no user energy saving policy) in terms of mean application service delivery delay, economic cost, energy expenditure, successful task completion ratio, throughput ratio, service provider total profit, user total benefit ratio or welfare, network lifetime value, and user survivability ratio.

The strengths and weaknesses of previous literature are presented in Section II. Section III describes the proposed network diagram, the liberty algorithm for resource management, and the operating method for running multiverse, metaverse, and non-metaverse-based applications. Section IV describes the mathematical model for the suggested liberty system, which includes multiple performance analysis metrics. Section V displays the suggested liberty scheme's performance analysis results and findings. Section VI provides a summary of the suggested liberty scheme outcomes.

II. Related Works

This section reviews some existing research on 6G, metaverse, MEC, and NFV technology. In [71], the authors conducted a survey on 6G technologies, applications, requirements, and future research challenges associated with 6G application execution. They specifically covered a variety of 6G applications, such as XR (extended reality)-based gaming or virtual tour applications, digital twin-based simulation applications, NFT-based art selling applications, and blockchain-based healthcare applications. The article in [72] surveyed the metaverse pipeline ecosystem, including its computing and networking structure, interaction layer, and digitization layer. They also talked about different technologies and the challenges associated with metaverse applications. The authors in [73] used metaverse and digital twin technology for smart manufacturing applications. The authors also mentioned key technological and development pillars for the metaverse, including blockchain, the internet of things (IoT), extended reality (XR), brain-computer interfaces, 3D reconstruction, and edge computing.

The authors in [74] emphasized the necessity of 6G technologies (terahertz or THz bands, blockchain, digital twin, open radio access networks) for proper service delivery, satisfying requirements, and efficient interaction in the metaverse. The article in [75] discussed energy-saving issues and challenges of implementing green metaverse networking. The authors of [76] discussed the role of next-generation wireless networks, collaborative robots (cobots), cyber-physical social systems, artificial intelligence, smart resource orchestration, softwarized networks, cloudification, and blockchain technologies in industry 5.0-based 6G application execution. The authors in [77] discussed several IoT-based metaverse applications that use the coordination of immersive cyber and physical worlds in different fields such as education, entertainment, healthcare, and smart cities. They talked about some notable metaverse applications, including virtual gaming, avatar-based socialization, concert viewing, blockchain-based virtual trading with NFT tokens, XR-based healthcare or surgery assistance, XR-based 3D apartment viewing, remote telepresence, and XR-based museum tours.

The article in [78] elaborated on the computation, caching, communication, and data-intensive network infrastructure, and the necessary requirements for MEC, SFC (service function chaining), and XR device-aided 6G metaverse app execution. In [79], the authors explored various principles, technologies, major components, attributes, scenarios, and applications for metaverse applications. They also investigated various communication, networking, security, and privacy issues related to distributed metaverse architecture. The article in [80] discussed the importance of some technologies such as network slicing (NS), next generation communication technologies (e.g., THz), XR technologies

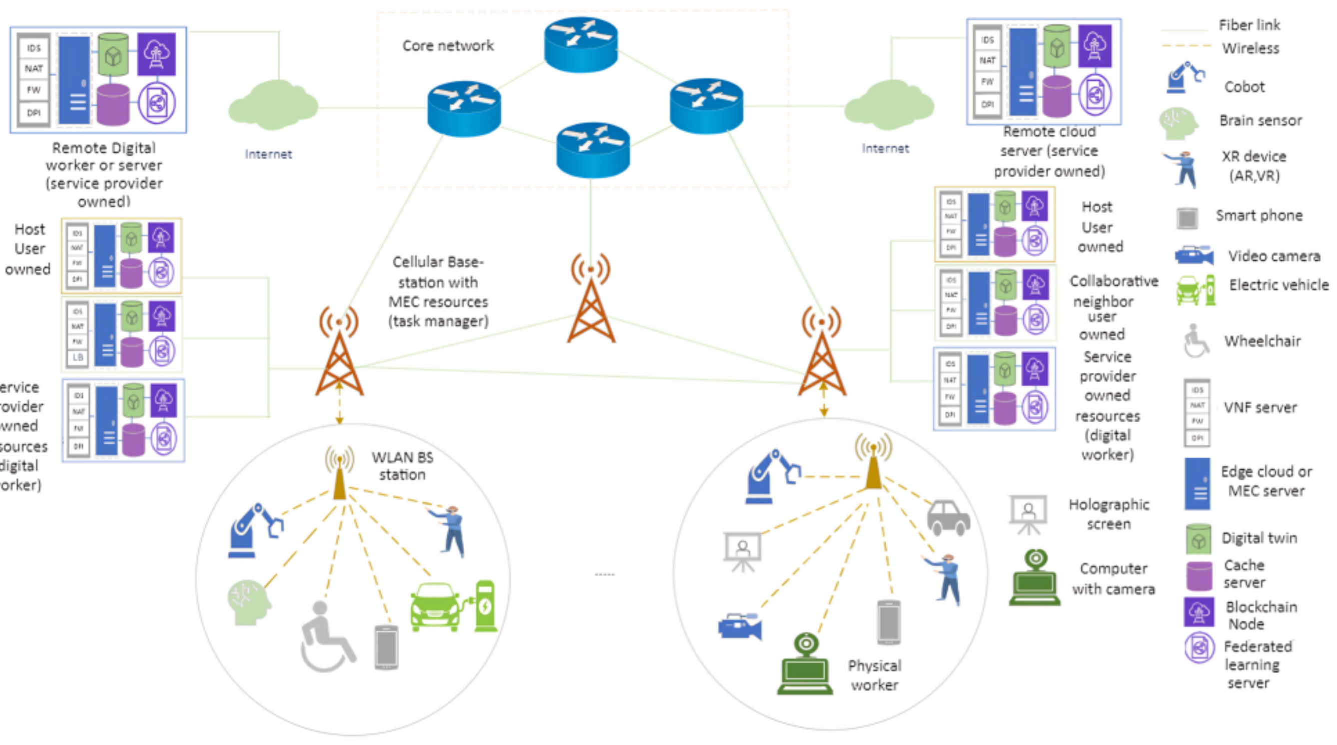


Fig. 1. Proposed 6G edge computing network for multiverse, metaverse, and non-metaverse application execution

(e.g., AR, VR, MR), IoT, robotics, blockchain, and multi-access edge computing (MEC) technologies for realizing some emerging metaverse applications. The work in [81] discussed security, energy, economy, and privacy threat issues for metaverse applications. The authors also presented a distributed metaverse network architecture using the interaction and coordination of human worlds, cyberworlds, and physical worlds. The article in [82] presented the metaverse's integration with edge intelligence for the execution of various 6G applications. The authors also investigated and highlighted three 6G edge intelligence-based Metaverse solutions, as well as the research challenges they faced. In [21], a joint optimization problem was developed to reduce energy consumption and delay during metaverse application execution. This included metaverse task splitting, partial computation offloading, and caching. To find the best offloading solution, the authors employed a double-deep Q-network. According to the authors of [83], the machine learning-based prediction system can be used for various metaverse applications, including avatar modeling, wireless network air interfaces, and resource scheduling. In [1], a hybrid fog-cloud collaboration-based network architecture was proposed to meet latency requirements for various metaverse applications. The authors also discussed the various network components for physical, virtual, and social space in the metaverse. In [84], the authors discussed economic system framework challenges, incentive mechanisms, future targets, the monetary system, security protocols, cross-chain technologies and protocols, basic considerations, and fundamentals for current and future metaverse applications. The article in [85] presented six evolutionary characteristics of the metaverse, taking into account users' perspectives such as economic system, civilization, immersion, low latency, real-time synchronization, and identity. The authors in [86] provided a detailed survey on 6G networks (e.g., terrestrial, non-terrestrial, space networks) along with different requirements (e.g., massive ultra-reliability, ultra-low latency, ubiquitous mobile broadband) associated with 6G applications and enabling technologies (e.g., blockchain, terahertz, MEC, digital twin, metaverse XR technologies) for 6G application execution. In [87], the authors provided a comprehensive review regarding IoT-enabled 6G applications such as autonomous driving, brain-computer interaction, holographic communication, blockchain and digital twin-based smart healthcare, XR-based education or training, and industrial manufacturing applications, among others. The authors classified applications based on their requirements, including further FeMBB, ELPC, eURLLC, umMTC, and LDHMC. In [88], the authors proposed a privacy-aware machine learning scheme for avatar-based NFT trading in the metaverse. The article in [89] suggests combining blockchain and 6G technologies to meet AR/VR application requirements. The works in [90] illustrated some unique challenges associated with the 6G-enabled metaverse, such as resource orchestration, service requirements satisfaction, collaboration and synchronization, and proper network and communication infrastructure for dynamic multi-user XR services, among others. In [91], the authors proposed a blockchain-based federated learning framework to improve the security and transparency of model learning in the metaverse. The authors in [92], [93], [94], [95], and [96] examined the suitability of different 5G and 6G technologies for different IoT application executions, including blockchain, mURLLC, MEC-enabled edge intelligence, reflecting interference surfaces (RISs), space-air-ground-underwater communications (SAGIN), and terahertz communication technology (THz). The authors of [93] discussed the potential of blockchain technologies (e.g., audit of network resources, securing information exchange, improving user trust) for different human-centric, machine-centric, and data-centric 6G applications and some research issues associated with blockchain, such as the selection of suitable blockchain types, key exchange protocols, and consensus algorithms for different 6G applications.

According to [97], MEC-based computation offloading is a promising technique for meeting the latency and energy requirements of various AR/VR-based metaverse applications. In [98], the authors developed a mixed integer non-linear programming (MINLP)-based optimization problem for virtualizing nework functions in the metaverse to reduce resource costs and optimize resource allocation. The authors of [99] discussed the main communication and networking challenges associated with immersive edge-enabled metaverse application execution, as well as some solutions. In [100], a metaverse architecture using mobile edge computing, digital twin, and blockchain technologies was proposed for real-time interaction between virtual and physical worlds. The authors also identified some challenges for metaverse-based application execution, such as proper offloading and caching, local machine learning model training, edge resource optimization policies, and incentive mechanisms for metaverse stakeholders. The works in [101] highlight the challenges and necessity of privacy-preserving techniques in metaverse applications. The article in [102] presented a survey regarding network slicing management techniques for industrial IoT applications, including various 5G and B5G applications (e.g., transportation, factory automation). [103] used a Q-learning algorithm to predict virtual network traffic load in fiber-wireless networks. Next, this paper will review the literature on VNF execution in the NFV-enabled network. According to [104], NFV technology has the potential to replace traditional hardware-based network function execution (e.g., load balancers, firewalls, network address translations or NATs, digital packet inspection or DPIs, and intrusion detection system or IDS) with software installations in a virtualized setup. In the NFV, a series of virtual network functions, or VNF (the execution of service function chaining), can run on generic servers to implement the desired service using virtualization technology. In [104], the authors created a non-linear optimization problem for a delay-sensitive VNF scheduling problem and solved it using a heuristic approach. To reduce energy consumption and resource costs in blockchain-enabled NFV networks, the authors in [105] proposed a hungarian-based server allocation for VNF processing. Some works in this domain focused on the energy efficiency and bandwidth usage minimization issue [106], [107], [108], [109].

According to [110], vehicular networks require softwarization and virtualization of network functions enabled by machine learning. In [111], the authors outlined a survey that includes the challenges and research issues associated with SFC (chain of network service function execution) in 5G and B5G networks, such as suitable server selection for VNF execution, path selection and routing, service function placement, QoS satisfaction, security and privacy issues, among others. In [112], the authors used a mixed-integer linear programming-based optimization problem and a heuristic algorithm to solve the VNF placement problem in an NFV-cloud-enabled network. To reduce energy consumption in software defined networking (SDN) and NFV-enabled networks, the authors in [113] utilized an NSGA-based evolutionary algorithm for the VNF deployment problem. In [114], the authors proposed using NFV technology to address content retrieval, cache placement, and selection issues, allowing for the coexistence of information centric networking (ICN) and internet protocol (IP) services. To reduce latency and resource usage, the authors of [115] used a breadth-first search-based SFC deployment algorithm for NFV-enabled services.

In [116], the authors used a genetic algorithm for server selection to address the issue of service function chaining placement in NFV-enabled video surveillance applications. The article in [117] provided a systematic overview of NFV data centers, covering research issues, potential technologies, architectures, and performance analysis tools, among others. In [118], the authors proposed a deep reinforcement learning-based solution for VNF placement in the O-RAN system to reduce computational costs and communication overheads. In [119], the authors emphasized the importance of an AI-based network slicing solution for 6G networks, including service demand prediction, VNF placement, resource orchestration, and VNF server selection. In [120], the authors presented a converged network architecture model for resource slicing in SDN and NFV-enabled optical-wireless networks. In [121], the authors used the first-come-first-served (FCFS) algorithm for multicast request admission in NFV-enabled mobile edge cloud networks to meet delay tolerance and cost-saving requirements. To overcome the overload problem of VNF deployment, the authors in [122] suggested using an automatic migration scheme for VNFs in cloud networks. The authors of [123] created a resource-aware physical node selection and FCFS-based joint SFC embedding and scheduling mechanism for NFV-enabled 6G networks. The article in [124] created an application-aware VNF deployment and SFC embedding policy for hybrid NFV-enabled networks. The work in [125] discussed a SLA and energy-aware VNF execution policy in NFV-enabled 5G networks. In [126], the authors formulated a reliability-based VNF placement optimization problem using integer linear programming for SFC execution in NFV-enabled mobile-edge computing networks. In [127], a survey was conducted to identify anomalies in NFV services and methods for detecting them. In [128], an optimization problem was developed to optimize network energy savings for joint user association, routing traffic, and VNF job placement across 6G heterogeneous networks. In [129], the authors presented a service chain re-routing scheme to achieve load balancing for NFV work execution by formulating an integer linear programming (ILP)-based optimization problem. The articles in [130] and [131] formulated an integer non-linear programming-based problem and presented a heuristic-based algorithm for SFC (ordered execution of VNF) planning over Space-Air-Ground Integrated (SAGIN) networks. Some works in the literature have focused on IoT technology, offloading, cloud computing, and communication medium issues for 5G and 6G applications [132], [133], [134], [135], [136], [137], [138], [139], [140], [141], [142].

The articles in [143] and [144] investigated the virtual network function (VNF) server deployment problem using genetic algorithms and Q-learning methods, respectively.

The existing literary article discussion reveals that they did not investigate any resource management scheme for both VNF server, physical worker, and digital worker selection while taking into account multiple multiverse applications, metaverse applications, and non-metaverse apps. They also did not look into the three-way collaboration and use of user-owned, neighbor-owned, and service provider-owned resources. Unlike previous articles, this paper presents an economy cost, energy cost, time criticality of applications, three-fold collaboration (use of user-owned, neighbor-owned, and service provider-owned resources), and network function virtualization-aware resource management scheme for multiverse, metaverse, and non-metaverse application execution over 6G edge computing networks.

III. Proposed Liberty-based Resource Management Scheme for Multiverse Application Execution over 6G Networks

A. Overview of the Proposed Network and Considerations

Figure 1 depicts the proposed 6G-based network model for multiverse, metaverse, and non-metaverse application execution using the liberty scheme. The systematic network model is divided into four sub-parts. In the first sub-part (access network), user devices connect to the network via a cellular base station and a WLAN (Wireless local area network) access point-based wireless communication link. The access networks' user devices include XR devices (users with extended reality devices, hololens, VR goggles, and haptic devices), cobots (collaborative robots), electric vehicles, humans in wheelchairs, video camera devices, humans with brain sensors, holographic projectors, computers, smart phones, and IoT devices, among others.

The second part of this network is an edge computing system. The second sub-part describes the edge computing system, which includes the virtual network function processing server (VNF server), edge computing servers or MEC servers, cache servers, digital twins, blockchain nodes, and federated learning servers deployed near the celluar base station. The task manager is positioned at the cell base station. The task manager can collect user multiverse, metaverse, and non-metaverse application requests and assign appropriate physical and digital worker and communication resources to each application processing.

The edge computing system has three types of resources: user-owned resources, collaborative neighbor-owned resources, and service provider-owned resources (such as MEC servers, caching servers, and VNF processing servers). The task manager selects the best suitable work nodes and resources for user application request processing based on the proposed resource scheduling scheme, evaluating resources owned by users, neighbors, and service providers. The network edge computing digitalworker nodes and servers are linked to the task manager or cellular base station via a dedicated optical fiber connection. The task managers at the cellular base stations can communicate with one another via a dedicated optical fiber communication link.

The third part of the network is the core network, which connects multiple routers via a fiber-based communication link. The task manager device at the cellular BS is linked to the core network using a dedicated fiber connection. In the fourth sub-part, the remote or central computing server is situated near the core network. A fiber link connects the core network to the remote cloud computing system. The remote cloud computing unit or digital server node can be located three to five times farther away than the edge computing digital server system.

In the network access sub-part, user devices can access

Algorithm 1 Proposed Liberty Algorithm

```
 1: for task manager device at cellular base station do
 2:     conveys elementary beacon to user devices, receives user registrations, application registrations, and network connectivity requests from users, transfers manager responses to user regarding registration and connectivity requests.
 3:     conveys the application request transfer timeslot schedule message to users, receives users application execution request during assigned control slot
 4:     conveys worker device resource status request, receives resource update response (worker), dispatches/receives other task managers schedule request/response messages.
 5:     reorders metaverse, multiverse, and non-metaverse aplication requests into three group ultra high, medium, and low time critical application group.
 6:     if application request==ultra-high critical then
 7:         executes prior to the medium/low time critical applications, reorder each ultra-high application based on small deadline first. checks all user-owned, neighbor-owned, and service provider-owned resources.
 8:         reserves the best VNF server with min VNF processing delay ($\beta^k_{pvp/svp/pvpp}$), best physical worker with min physical work processing delay ($\beta^k_{xpu/xpr/xpuu/xpuc/xpus/xmu/xpuh}$), best digital worker device (cloud/caching/digital twin/BC/FL server) with min digital work processing delay ($\beta^k_{dwp/dwpp/bcc/bvm/dwps/dwop/csp}/$), best communication resource with minimum data dispatch delay ($\beta^k_{xdu/rdt/rdtt/xduu/tpd/rdu/btt/rdtc}$) for each re-ordered application with a min predicted delay $\beta^k_{sdp}$, min $G^k_{ee}$, and min $\gamma^k_{uec}$
 9:     else if application request==medium critical then
10:         executes before the low time critical and after the ultra-high time critical application group. reorders each medium critical applications based on small deadline application first.
11:         reserves the best VNF resources, physical, digital resources, communication resources for each medium-critical app with a min predicted delay $\beta^k_{sdp}$, min $G^k_{ee}$, and min $\gamma^k_{uec}$
12:     else
13:         executes after ultra-high and medium-critical applications. re-orders all remaining low-critical applications based on their small deadline first.
14:         allocates and reserves the best VNF resources, physical, digital resources, communication resources for each low critical app with a min predicted delay $\beta^k_{sdp}$, min $G^k_{ee}$, and min $\gamma^k_{uec}$
15:     end if
16:     dispatches the resource schedule message to users.
17: end for
```

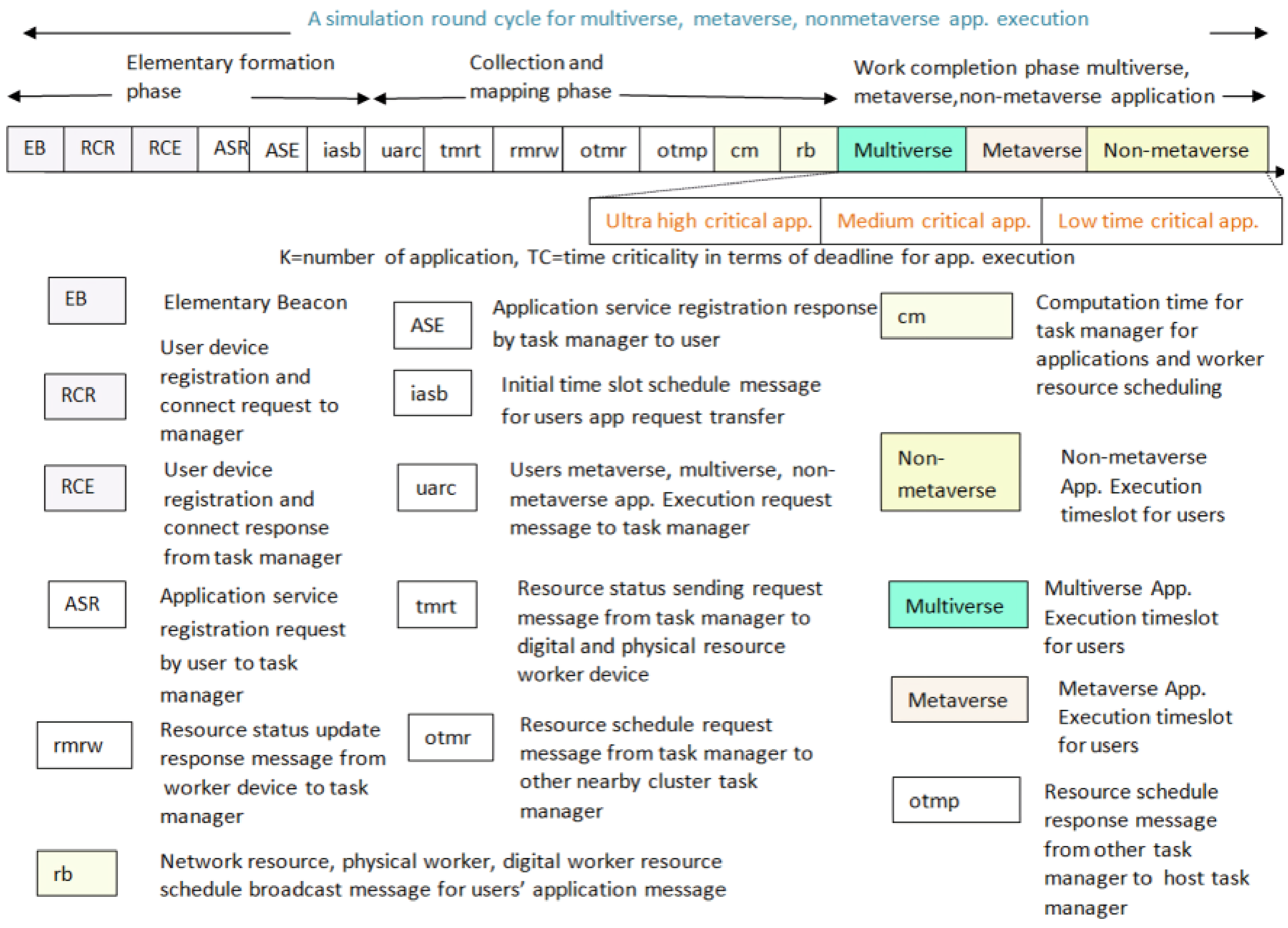

A simulation round cycle for multiverse, metaverse, nonmetaverse app. execution
Elementary formation phase
Collection and mapping phase
Work completion phase multiverse, metaverse,non-metaverse application
EB
RCR
RCE
ASR
ASE
iasb
uarc
tmrt
rmrw
otmr
otmp
cm
rb
Multiverse
Metaverse
Non-metaverse
Ultra high critical app.
Medium critical app.
Low time critical app.
K=number of application, TC=time criticality in terms of deadline for app. execution
EB Elementary Beacon
RCR User device registration and connect request to manager
RCE User device registration and connect response from task manager
ASR Application service registration request by user to task manager
rmrw Resource status update response message from worker device to task manager
rb Network resource, physical worker, digital worker resource schedule broadcast message for users' application message
ASE Application service registration response by task manager to user
iasb Initial time slot schedule message for users app request transfer
uarc Users metaverse, multiverse, non-metaverse app. Execution request message to task manager
tmrt Resource status sending request message from task manager to digital and physical resource worker device
otmr Resource schedule request message from task manager to other nearby cluster task manager
cm Computation time for task manager for applications and worker resource scheduling
Non-metaverse Non-metaverse App. Execution timeslot for users
Multiverse Multiverse App. Execution timeslot for users
Metaverse Metaverse App. Execution timeslot for users
otmp Resource schedule response message from other task manager to host task manager

Fig. 2. Proposed work timing or sequence model for multiverse, metaverse, and non-metaverse application execution

internet facilities through different wireless links such as tera-hertz or THz links (i.e., by following the IEEE 802.15.3.d standard with a .7 THz bandwidth amount and link distance range of 1–15 m), traditional microwave links (by following the IEEE 802.11b standard with a 6.2 GHz bandwidth amount and link range within 1-200 m), and millimeter wave or mm-wave links (by following the IEEE 802.11ad standard with a 1.25 GHz bandwidth amount). Furthermore, user devices can connect to the internet via WiFi access points (using the IEEE 802.11ad standard with a 1.25 GHz bandwidth amount). Furthermore, user devices can connect to the internet via WiFi access points (using the IEEE 802.11b standard, 6.2 GHz bandwidth amount, and link range within 1-200 m) and millimeter wave or mm-wave link (by following the IEEE 802.11ad standard, 1.25 GHz bandwidth amount. Furthermore, user devices can connect to the internet via WiFi access points (using the IEEE 802.11b standard, with a bandwidth of 7 GHz and a range of 1-100 m).

The IEEE 802.3 cd standard governs the fiber-based communication link between the edge computing system and the cellular base station, as well as the cellular base station's connection to a remote cloud server via a core network. The cache server in the edge computing system uses zipfs law-based cache distribution [141] and the least recently used (LRU) cache replacement policy [142]. The VNF server on the edge computing system can provide a variety of virtual network function processing services, including firewalls (FW), intrusion detection systems (IDS), load balancing (LB), digital packet inspection (DPI), and network address translation (NAT). The blockchain node, federated learning (FL)-based task processing server, metaverse-related task processing server, and digital twin-based work processing server are located within the edge computing server, or MEC server (digital worker node).

In this work, the digital work node resource can be owned by both host users and collaborative neighbor users, as well as service providers. In this work, triple collaboration occurs between host-owned resources, neighbor-owned resources, and service provider-owned resources. As a result, the task manager can choose the best digital worker for users of various applications by considering all three types of digital worker resource ownership. The maximum computational work processing speed (CPU speed) for a digital worker device (MEC server, digital twin server, FL server, VNF server, blockchain device) is 4.5 GHz. The user device (XR device, smartphone, cobot, brain sensors, video camera, vehicle, hololens) has a maximum CPU speed of 2000 MHz. During the request timeslot period, the user device can send the task manager a multiverse, metaverse, or non-metaverse application request. After receiving all user application requests, the task manager assigns communication resources, as well as physical and digital worker resources, to each request. In this work, applications are classified according to their time-criticality. The task manager can allocate communication, physical, and digital resources based on an ultra-critical application first basis (applications with a short deadline for execution), as well as high service delivery delay, energy gain, and economic cost gain. During each user work completion timeslot, the user device, physical work node, and digital work node process the application workload (multiverse, metaverse, and non-metaverse applications) and finish the result generation. The user device and physical worker device can capture and upload application data to a digital work node for further processing and output. The digital work node processes the offloaded application data and returns the application results to the users. The application execution sequence diagram is depicted in Figure 2.

B. Liberty-based resource management scheme

This paper will then present the proposed liberty-based resource management scheme for metaverse, multiverse, and non-metaverse application execution over 6G networks. The liberty-based resource management algorithm is discussed in Algorithm 1. Figure 2 displays the resource management and application work completion timing sequence model. The simulation time cycle, or round, is divided into three parts: the elementary formation phase, the collection and mapping phase, and the application work completion phase. During the elementary phase (first phase), the task manager starts the time cycle by sending the elementary beacon message (EB) to the user devices (such as XR devices, cobots, smartphones, and vehicles).The user devices send the RCR message (user registration and network connectivity request) and the ASR (application service registration request) to the task manager at the cellular base station device. The task manager responds with an RCE (user registration and network connectivity response) and an ASE (application service registration response) message, which are both delivered to the user devices. The task manager then sends the initial time slot schedule message (iasb) to the user devices.

Next, the collection and mapping phases will begin. The user device then sends the application request message (which includes multiverse, metaverse, and non-metaverse) to the task manager during the designated user application request dispatch (uarc) time slot. The task manager then sends the resource status update request (tmrt) to both physical and digital worker devices. The task manager, in turn, receives resource update response (rmrw) messages from worker devices (resources owned by the user, neighbor, or service provider).

Unlike previous works, we considered collaboration and cooperation among user-owned, neighbor-owned, and service provider-owned devices for resource usage in multiverse, metaverse, and non-metaverse application execution. That is, application processing resources can be chosen from user-owned, neighbor-owned, and service provider-owned resources (such as digital work processing nodes, VNF servers, MEC servers, and physical devices).

The host task manager also communicates with the other nearby task manager to obtain their resource schedule information via resource schedule request (otmr) and response (otmp) messages. The host task manager then calculates the resource selection and scheduling process during the computation period (cm). During this time, the task manager reorders all application requests (e.g., multiverse, metaverse, and non-metaverse applications) into three groups based on lower application execution deadlines: high-time critical, medium-time critical, and low-time critical. Following the completion of high-time critical applications, medium- and low-time critical applications are executed. The application with the highest time criticality will be executed before the medium and low time critical application groups. All applications in each group are rearranged and served (assigned resources) in accordance with their small application execution deadline value.

Before each application work execution phase, the task manager recommends the best physical worker device, digital worker device (cloud/caching/digital twin/BC/FL server), VNF processing server, and communication link resource selection based on the minimum predicted application work completion delay (β^k_{sdp}). First, the task manager reorders the high-time critical application request based on the small deadline application first order. Then, the task manager reserves and schedules the best VNF server with min VNF processing delay (min $\beta^k_{pvp/svp/pvpp}$), best physical worker with min physical work processing delay (min $\beta^k_{xpu/xpr/xpuu/xpuc/xpus/xmu/xpuh}$), best digital worker device with min digital work processing delay (min $\beta^k_{dwp/dwpp/bcc/bvm/dwps/dwop/csp}$), best communication resource with minimum data dispatch delay ($\beta^k_{xdu/rdt/rdtt/xduu/tpd/rdu/btt/rdtc}$) for each re-ordered high-time critical application with min predicted work completion delay (β^k_{sdp}), min predicted energy expenditure for users (G^k_{ee}), and min predicted economic cost for users (γ^k_{uec}).

Next, the task manager reorders each medium-critical application so that the small deadline application comes first. Then, the task manger checking the remaining available resources and reserves the best VNF server with min VNF processing delay ($\beta^k_{pvp/svp/pvpp}$), best physical worker with min physical work processing delay ($\beta^k_{xpu/xpr/xpuu/xpuc/xpus/xmu/xpuh}$), best digital worker device with min digital work processing delay ($\beta^k_{dwp/dwpp/bcc/bvm/dwps/dwop/csp}/$), best communication resource with minimum data dispatch delay ($\beta^k_{xdu/rdt/rdtt/xduu/tpd/rdu/btt/rdtc}$) for each medium critical application with a with min predicted work completion delay (β^k_{sdp}), min predicted energy expenditure for users (G^k_{ee}), and min predicted economic cost for users (γ^k_{uec}).

Next, the task manager re-orders all remaining low-critical applications based on their small deadline first order. The task manager allocates and reserves the resources by checking all remaining available resources. The task manager selects best VNF resources min VNF processing delay (minimum $\beta^k_{pvp/svp/pvpp}$), best physical worker with minimum physical work processing delay (minimum $\beta^k_{xpu/xpr/xpuu/xpuc/xpus/xmu/xpuh}$), best digital worker device with minimum digital work processing delay ($\beta^k_{dwp/dwpp/bcc/bvm/dwps/dwop/csp}$), best communication resource with minimum data dispatch delay (min $\beta^k_{xdu/rdt/rdtt/xduu/tpd/rdu/btt/rdtc}$) for each low critical app with a min predicted work completion delay (β^k_{sdp}), min predicted energy expenditure for users (G^k_{ee}), and min predicted economic cost for users (γ^k_{uec}).

After resource reservation and mapping are completed, the task manager sends the network VNF resource server, physical worker, digital worker resource node, and communication link resource scheduling message (rb message) to all workers and users. Finally, the work completion phase (third phase) completes the metaverse, multiverse, and non-metaverse application execution work completion process in the order of their resource scheduling (i.e., ultra high critical application first, then medium critical application execution, and finally low critical application execution).

During a metaverse application execution time slot (e.g., XR-based immersive gaming), the user device first needs to connect to the internet and the metaverse platform. After that, the immersive gaming application request is dispatched to the task manager. The task manager device next processes the application request and delivers the work completion instruction to workers (players device, digital worker, physical worker, VNF server). The users or players XR device next captures the gaming information (e.g., action and state associated with user game strategy, sound and image, input data, moving, shooting, searching action information) and uploads the selected game server (digital worker or MEC server). Before the user's game input data is offloaded, the chosen VNF server performs the virtual network function processing operation (for example, firewall, intrusion detection, load balancing, digital packet inspection, and network address translation). Following that, the processed data from the VNF server is routed to a specific digital worker or MEC-based gaming server for processing. Based on the user's input, the digital worker processes game information and generates the game's next state or result (e.g., final environment generation, user game point data generation, next frame or output frame production, immersion, static data addition, rendering frame, computation). The digital worker processes game data before sending it to the user's XR device for display. However, before the output gaming data is delivered to the receiver's device, the selected VNF server performs secondary virtual network function processing (e.g., intrusion detection, firewall, and network address translation) and sends the final resultant data to the users. Finally, users receive the final gaming application output data and have the immersive gaming experience displayed to them via XR devices.

Similarly, during a non-metaverse application execution time slot (for example, a brain-computer-wheelchair device interaction application), the user device first connects to the internet before sending their application request to the task manager. The task manager handles application requests and sends application instructions to user and worker devices (such as brain sensors, XR devices, wheelchairs, MEC servers, and VNF servers). Next, the user's brain sensor devices collect brain sensing data (via MRI or CT scan) and send it to the selected digital worker's MEC device for processing.

Before the user's brain sensor-computer-wheelchair interaction application input data is offloaded, the chosen VNF server performs the virtual network function processing operation (for example, firewall, intrusion detection, load balancing, digital packet inspection, and network address translation). Following that, the VNF server processes the data and sends it to a designated digital worker or MEC server. The digital work node executes and generates the application result (for example, the brain sensor instructs the wheelchair to move or not). Next, before delivering the results to the wheelchair device, the selected VNF server performs secondary virtual network function processing (e.g., intrusion detection, firewall, network address translation) and sends the final resultant data to the user's wheelchair device. The MEC server sends the final result instruction to the user's wheelchair device, which then executes the brain instruction (for example, moving the wheelchair based on brain signal data).

During the multiverse application execution time slot, the respective user devices, VNF server, physical worker, and digital worker complete their own tasks before delivering the application results to the users/receivers devices. In this work, the multiverse application (avatar-based socialization and NFT-based virtual art selling) is the result of multiple metaverse platform-based work executions. For example, avatar interaction in the metaverse is one type of work, while virtual art trading via NFT (non-fungible token) on the metaverse platform is another. Section 4 discusses the working steps for avatar interaction and NFT-based trading applications.

IV. Analytical Model

This section presents the analytical model that supports the proposed liberty scheme. The performance metrics examined are mean application service delivery delay, economic cost for users, successful task completion ratio, throughput, profit for service providers, users' total benefit, unused energy, network lifetime, and survivability ratio.

A. Mean Application Service Delivery Delay

The end-to-end application service delivery delay (β^k_{asd}) for multiple application execution includes not only elementary network formation phase delay (β^k_{ep}) but also both information collection and resource mapping phase delay (β^k_{sp}), and different application (i.e., multiverse, metaverse,and non-metaverse application execution) work completion phase delay ($\beta^k_{sdp} = \beta^k_{mv} + \beta^k_{me} + \beta^k_{nme}$). The mean application service delivery delay (β^k_{mad}) for k number of applications ($\beta^k_{mad} = \frac{\sum_{k=1}^{z} \beta^k_{asd}}{k}$) is analyzed by:

$$\beta^k_{mad} = \frac{\sum_{k=1}^{z} \beta^k_{ep} + \beta^k_{sp} + \beta^k_{sdp}}{k} = \frac{\sum_{k=1}^{z} \beta^k_{ep} + \beta^k_{sp} + \beta^k_{mv} + \beta^k_{me} + \beta^k_{nme}}{k} \quad (1)$$

Where k is the total application number. β^k_{mv}, β^k_{me}, and β^k_{nme} are the multiverse, metaverse, and non-metaverse application work completion delay, respectively.

The first elementary phase delay includes task manager-based beacon transfer to user devices, user device registration and connectivity with the network, application service registration, and an initial control slot assignment schedule for user application request transfer. The elementary phase delay

β_{ep}^{k} is ascertained by:

$$\beta_{ep}^{k} = \frac{\Psi_{eb}^{k} + \Psi_{rcr}^{k} + \Psi_{rce}^{k} + \Psi_{asr}^{k} + \Psi_{ase}^{k} + \Psi_{iasb}^{k}}{J_{wr}} * u_{wr} + \frac{\Psi_{eb}^{k} + \Psi_{rcr}^{k} + \Psi_{rce}^{k} + \Psi_{asr}^{k} + \Psi_{ase}^{k} + \Psi_{iasb}^{k}}{J_{or}} * u_{or} + \frac{Q_{pd}^{ep}}{\Lambda_{pd}} + \frac{Q_{tm}^{ep}}{\Lambda_{dd}} + \beta_{pp}^{k} + \beta_{rwd}^{k} \quad (2)$$

$\Psi_{eb}^{k}, \Psi_{rcr}^{k}, \Psi_{rce}^{k}, \Psi_{asr}^{k}, \Psi_{ase}^{k}$, and Ψ_{iasb}^{k} are the primary beacon message data size (by task manager to users), network registration request message size (by user device to task manager), network registration response message size (by task manager to user device), 6G application service (metaverse, multiverse, non-metaverse) registration request message (by user device to task manager), application service registration response message size (by task manager to user device), and initial time-slot assign schedule message size (by task manager to user device), respectively.

Q_{pd}^{ep} is the workload for the user device during the elementary phase for different request message generation. Q_{tm}^{ep} is the workload for task managers during the elementary phase for different response message generation and timeslot schedule message generation for users. Λ_{pd} and Λ_{dd} are workload processing speeds for user devices and task managers (digital workers at MEC), respectively.

J_{wr}, J_{or}, u_{or}, and u_{wr} are the wireless link-based data exchange rate, fiber-link-based data exchange rate, hop number during optical fiber-link-based application data transfer, and hop number during wireless link-based application data transfer, respectively. β_{pp}^{k} is the propagation delay latency and β_{rwd}^{k} is the resource access queuing latency. The resource access waiting and queuing delay is measured by taking into account the well-known M/D/1 queuing delay calculation model as given in [132].

The second phase is the information collection and resurce mapping phase (second phase). The resource and application information collection along with the resurce mapping phase (second phase) delay Δ_{urg}^{i} is investigated by:

$$\beta_{sp}^{k} = \frac{\Psi_{uarc}^{k} + \Psi_{tmrt}^{k} + \Psi_{rmrw}^{k} + \Psi_{otmr}^{k} + \Psi_{otmp}^{k}}{J_{wr}} * u_{wr} + \frac{\Psi_{uarc}^{k} + \Psi_{tmrt}^{k} + \Psi_{rmrw}^{k} + \Psi_{otmr}^{k} + \Psi_{otmp}^{k}}{J_{or}} * u_{or} + \frac{Q_{pd}^{sp}}{\Lambda_{pd}} + \frac{Q_{dd}^{sp}}{\Lambda_{dd}} + \frac{\Psi_{urb}^{k} + \Psi_{wrb}^{k}}{J_{wr}} * u_{wr} + \frac{\Psi_{urb}^{k} + \Psi_{wrb}^{k}}{J_{or}} * u_{or} + \frac{Q_{cm}^{sp}}{\Lambda_{dd}} + \beta_{pp}^{k} + \beta_{rwd}^{k} \quad (3)$$

$\Psi_{uarc}^{k}, \Psi_{tmrt}^{k}, \Psi_{rmrw}^{k}, \Psi_{otmr}^{k}$, and Ψ_{otmp}^{k} are the data sizes regarding user application requests (from user device to task manager at the edge network), resource status request message sizes (from task manager to worker device), resource status response message sizes (from worker to task manager), resource schedule information requests to other nearby task managers (from host task manager), and resource schedule information transfers to host task managers (from other nearby task managers), respectively. Q_{pd}^{sp} is the workload associated with the user device for different message transfers during the second phase. Q_{dd}^{sp} is the workload associated with the task manager and digital worker device for different message transfers during the second phase. Q_{cm}^{sp} is the task manager workload count for resource schedule and application execution timeslot assignment work. Ψ_{urb}^{k} and Ψ_{wrb}^{k} are the resource schedule and application execution timeslot assignment message size for user device and worker device, respectively. Λ_{dd} is the task manager device workload processing speed.

The third phase is a different application work completion phase. The total application (multiverse, metaverse, and non-metaverse) work completion phase delay β_{sdp}^{k} is determined by:

$$\beta_{sdp}^{k} = \sum_{k=1}^{z} (\beta_{ig}^{k} + \beta_{mt}^{k} + \beta_{vce}^{k} + \beta_{mh}^{k} + \beta_{sa}^{k} + \beta_{hrp}^{k} + \beta_{fs}^{k} + \beta_{bnas}^{k} + \beta_{drs}^{k} + \beta_{ia}^{k} + \beta_{av}^{k} + \beta_{tb}^{k} + \beta_{vc}^{k} + \beta_{bcx}^{k} + \beta_{vet}^{k} + \beta_{snt}^{k} + \beta_{cmt}^{k}) \quad (4)$$

$\beta_{ig}^{k}, \beta_{mt}^{k}, \beta_{vce}^{k}, \Theta_{mh}^{k}, \beta_{sa}^{k}, \beta_{hrp}^{k}, \beta_{fs}^{k}, \beta_{bnas}^{k}$, and β_{drs}^{k} are the work completion delays for (metaverse applications) XR-based immersive gaming experiences, XR-based virtual museum tours, XR-based virtual clothing experiences, metaverse-based healthcare, metaverse-based avatar socialization, hologram-based remote presence, FL-based surveillance applications, NFT- and blockchain-based visual art sales, and DT-based robot worker selection for manufacturing, respectively.

$\beta_{ia}^{k}, \beta_{av}^{k}, \beta_{tb}^{k}, \beta_{vc}^{k}, \beta_{bcx}^{k}$, and β_{vet}^{k} are the work completion delays for (non-metaverse applications) industrial automation, autonomous vehicle applications, traditional broadband-based data transfer applications, video file caching, brain-computer-hololens-based interactive applications, and vehicle energy transfer applications, respectively.

β_{snt}^{k} and β_{cmt}^{k} are the work completion delays for (multiverse or multiple metaverse platform-based applications) avatar socialization with the NFT trading application and XR-based clothing experience with the virtual museum tour experience, respectively.

This paper first presents the analytical model to calculate the work completion delay for XR-based immersive gaming in the metaverse. The work completion delay for the XR-based immersive gaming application β_{ig}^{k} is highlighted by:

$$\beta_{ig}^{k} = \sum_{k=1}^{z} (\beta_{cim}^{k} + \beta_{trd}^{k} + \beta_{tmp}^{k} + \beta_{sip}^{k} + \beta_{xpu}^{k} + \beta_{xdu}^{k} + \beta_{pvp}^{k} + \beta_{dwp}^{k} + \beta_{rdt}^{k} + \beta_{svp}^{k} + \beta_{xpr}^{k}) \quad (5)$$

β_{cim}^{k} is the user or XR device initial connectivity delay with metaverse ($\beta_{cim}^{k} = \frac{q_{cim}}{\Lambda_{pd}}$). β_{trd}^{k} is the immersive gaming players application request dispatch delay to nearby task manager ($\beta_{trd}^{k} = \frac{\Psi_{ar}^{k}}{J_{wr}} * u_{wr} + \frac{\Psi_{ar}^{k}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$).

Where Ψ_{ar}^{k}, J_{wr}, J_{or}, u_{or}, and u_{wr} are the application request message size, wireless link-based data exchange rate, fiber-link-based data exchange rate, hop number during optical fiber-link-based application data transfer, and hop number during wireless link-based application data transfer, respectively. β_{pp}^{k} is the propagation delay latency and β_{rwd}^{k} is the resource access queuing latency. The resource access waiting and queuing delay is measured by taking into account the well-known M/D/1 queuing delay calculation model as given in [133]. q_{cim} and Λ_{pd} are the workload during internet connectivity and XR device internet connecting work speed, respectively.

β^{k}_{tmp} is the digital task manager (at the network edge or MEC)-based application initial instruction generation delay for the workers ($\beta^{k}_{tmp} = \frac{q^{k}_{tmp}}{\Lambda_{dd}}$). Where q^{k}_{tmp} is the number of workloads for the task manager regarding immersive gaming application instruction generation for the workers. Λ_{dd} is the application workload processing power at the digital task manager (MEC server).

β^{k}_{sip} is immersive gaming application work processing instruction transfer delay from the digital task manager (at the MEC) to workers ($\beta^{k}_{sip} = \frac{\Psi^{k}_{sip}}{J_{wr}} * u_{wr} + \frac{\Psi^{k}_{sip}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{k}_{sip} is the data size that includes application processing instructions for the workers.

β^{k}_{xpu} is the players XR device-based game information (game starting information, state, field of view, moving, shooting, attacking, sensing action selection) capture delay ($\beta^{k}_{xpu} = \frac{q^{ig}_{xpu}}{\Lambda_{pd}}$). q^{ig}_{xpu} is the XR device initial work instruction or workload for the immersive gaming application. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{k}_{xdu} is the immersive gaming player device captured gaming data upload delay to the digital worker device at the MEC server ($\beta^{k}_{xdu} = \frac{\Psi^{ig}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{ig}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{ig}_{di} is the data size that consists of the XR device captured immersive gaming state selection data or offloaded immersive gaming input data.

β^{k}_{pvp} is the primary virtual network function processing delay at the digital work node server for the immersive gaming application ($\beta^{k}_{pvp} = \frac{q^{k}_{atn} + q^{k}_{nfw} + q^{k}_{nid} + q^{k}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{k}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{k}_{atn}, q^{k}_{nfw}, q^{k}_{nid}, and q^{k}_{npdi} are the virtual network function workload amounts for network address translation (NAT), firewall (FW) operation, network intrusion detection or IDS operation, and offloaded packet digital detection (DPI) operation, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed).

Ψ^{k}_{tvpd}, J_{lr}, and u_{lr} are the exchanged data amounts during virtual network processing operations from one virtual network function processing server or node to another virtual network function processing node, transferred communication link associate data rate, and hop number during exchanged data dispatch from one link to another, respectively.

β^{k}_{dwp} is the digital work node (at the MEC serer) based offloaded game data processing (e.g., rendering frame, FoV, immersion, add static data, add sound, create new environments, next game state generation based on players action selection) and game result generation delay ($\beta^{ig}_{dwp} = \frac{q^{ig}_{dwp}}{\Lambda_{dd}}$). Where q^{ig}_{dwp} is the number of workloads for the uploaded immersive game data processing and result generation at the digital worker node. Λ_{dd} is the immersive game application workload processing power at the digital work processing node (MEC server).

β^{k}_{rdt} is the digital work node processed gaming resultant data dispatch delay to players XR device ($\beta^{k}_{rdt} = \frac{\Psi^{ig}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{ig}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{ig}_{rdt} is the output data size that consists of the immersive game application resultant data amount.

β^{k}_{svp} is the secondary virtual network function processing delay at the digital work node before the immersive gaming application output data reception at the players XR device. ($\beta^{k}_{svp} = \frac{q^{k}_{atn} + q^{k}_{nfw} + q^{k}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{k}_{stvd}}{J_{lr}} * u_{lr}$).

q^{k}_{atn}, q^{k}_{nfw}, and q^{k}_{npdi} are the virtual network function workload amounts for NAT, FW operation, and DPI operation, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{k}_{stvd} is the secondary exchanged data amount during a virtual network processing operation from one virtual network function processing server or node to another virtual network function processing node, respectively.

β^{k}_{xpr} is the receiver player XR device-based game output information (rendered frame with immersion) visualization and haptic sensation reception delay by haptic jacket ($\beta^{k}_{xpr} = \frac{q^{ig}_{xpr}}{\Lambda_{pd}}$). q^{ig}_{xpr} is the workload (gaming output data visualization with haptic sensation) for the receiver device (XR device and haptic jacket). Λ_{pd} is the work processing power or CPU speed for the user haptic jacket or XR device.

Next, this article will discuss the analytical model to calculate the work completion delay for an XR-based virtual museum tour application in the metaverse. The work completion delay for the XR-based virtual museum tour application β^{k}_{mt} execution is measured by:

$$\beta^{k}_{mt} = \sum_{k=1}^{z}(\beta^{mt}_{cim} + \beta^{mt}_{trd} + \beta^{mt}_{tmp} + \beta^{mt}_{sip} + \beta^{mt}_{xpu} + \beta^{mt}_{xdu} + \beta^{mt}_{pvp} + \beta^{mt}_{dwp} + \beta^{mt}_{rdt} + \beta^{mt}_{svp} + \beta^{mt}_{xpr}) \quad (6)$$

β^{k}_{cim} is the user or XR device initial connectivity delay with metaverse. β^{k}_{trd} is the virtual museum tour application that requests dispatch delay to the nearby task manager. β^{mt}_{tmp} is the digital task manager (at the network edge, or MEC)-based museum tour applications initial instruction generation delay for the workers. β^{k}_{sip} is virtual museum tour application work processing instruction transfer delay from the digital task manager (at the MEC) to workers. β^{mt}_{cim}, β^{mt}_{trd}, β^{mt}_{tmp}, and β^{mt}_{sip} are calculated by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see previous application discussion).

β^{mt}_{xpu} is the player's XR device-based QR code-based artifact and museum object image capture delay ($\beta^{mt}_{xpu} = \frac{q^{mt}_{xpu}}{\Lambda_{pd}}$). q^{mt}_{xpu} is the XR device initial work instruction or workload for the QR code-based artifact and museum object image capture work. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{mt}_{xdu} is the XR device captured QR code-based artifact and museum object image data upload delay to the digital worker device at the MEC server ($\beta^{mt}_{xdu} = \frac{\Psi^{mt}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{mt}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{mt}_{di} is the data size that consists of the XR device captured QR code-based artifact and museum object image data (i.e., offloaded input data).

β^{mt}_{pvp} is the primary virtual network function processing delay at the digital work node server for the metaverse-based virtual museum tour application ($\beta^{mt}_{pvp} = \frac{q^{mt}_{nfw} + q^{mt}_{nid} + q^{mt}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{mt}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{mt}_{nfw}, q^{mt}_{nid}, and q^{mt}_{npdi} are the virtual network function workload amounts for firewall, network intrusion detection, and packet digital inspection for virtual museum tour applications based on offloaded data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node.

Ψ^{mt}_{tvpd}, J_{lr}, and u_{lr} are the exchanged data amounts (XR device to digital worker MEC server) during virtual network processing operations from one virtual network function processing server or node to another virtual network function processing node, selected communication link data rate, and hop number during data dispatch from one link to another, respectively.

β^{mt}_{dwp} is the digital work node (at the MEC serer) based offloaded data processing (i.e., detect artifact picture, gather information, artifact environment setup, render image, add static data, show navigation path for museum tour, name of different museum object, produce museum tour data for viewing) and result generation delay ($\beta^{mt}_{dwp} = \frac{q^{mt}_{dwp}}{\Lambda_{dd}}$). Where q^{mt}_{dwp} is the number of workloads for the uploaded museum tour data processing and result generation at the digital worker node. Λ_{dd} is the virtual museum tour application workload processing power at the digital work processing node (MEC server).

β^{mt}_{rdt} is the digital work node processed virtual museum tour resultant data dispatch delay to players XR device ($\beta^{mt}_{rdt} = \frac{\Psi^{mt}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{mt}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{mt}_{rdt} is the output data size that consists of the virtual museum tour application resultant data amount.

β^{mt}_{svp} is the secondary virtual network function processing delay at the digital work node before the immersive gaming application output data reception at the players XR device. ($\beta^{mt}_{svp} = \frac{q^{mt}_{atn}+q^{mt}_{nfw}+q^{mt}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{mt}_{stvd}}{J_{lr}} * u_{lr}$).

q^{mt}_{atn}, q^{mt}_{nfw}, and q^{k}_{nid} are the virtual network function workload amounts for network address translation, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{mt}_{stvd} is the secondary exchanged data amount during virtual network processing operation during metaverse-based virtual museum tour application execution from one virtual network function processing server or node to another virtual network function processing node, respectively.

β^{mt}_{xpr} is the receiver player XR device-based virtual museum tour data result (rendered frame with immersion) visualization and haptic sensation reception delay by haptic jacket ($\beta^{mt}_{xpr} = \frac{q^{mt}_{xpr}}{\Lambda_{pd}}$). q^{mt}_{xpr} is the workload (virtual museum tour applications output data visualization with haptic sensation) for the receiver device (XR device and haptic jacket). Λ_{pd} is the work processing power or CPU speed for the user's haptic jacket or XR device.

The third application that we are going to discuss is an XR-based virtual clothing experience via the avatar. The work completion delay for XR-based virtual clothing experience β^{k}_{vce} execution is given by:

$$\begin{aligned}\beta^{k}_{vce} = \sum_{k=1}^{z}(&\beta^{vce}_{cim} + \beta^{vce}_{trd} + \beta^{vce}_{tmp} + \beta^{vce}_{sip} + \beta^{vce}_{xpu} + \beta^{vce}_{xdu} \\ &+ \beta^{vce}_{pvp} + \beta^{vce}_{dwp} + \beta^{vce}_{xpp} + \beta^{vce}_{xduu} + \beta^{vce}_{pvpp} + \beta^{vce}_{dwpp} \\ &+ \beta^{vce}_{rdt} + \beta^{vce}_{svp} + \beta^{vce}_{xpr}) \qquad (7)\end{aligned}$$

β^{vce}_{cim} is the initial internet connectivity delay for the user and XR device. β^{vce}_{trd} is the virtual clothing application request dispatch delay to nearby task manager. β^{vce}_{tmp} is the digital task manager-based virtual clothing experience applications initial instruction generation delay for the workers. β^{vce}_{sip} is virtual clothing application work processing instruction transfer delay from the digital task manager to workers. β^{vce}_{cim}, β^{vce}_{trd}, β^{vce}_{tmp}, and β^{vce}_{sip} are calculated by using the same calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see XR-based gaming application discussion).

β^{vce}_{xpu} is the user XR device-based person data (profile image for avatar, personal information) capture delay ($\beta^{vce}_{xpu} = \frac{q^{vce}_{xpu}}{\Lambda_{pd}}$). q^{vce}_{xpu} is the workload for XR device-based person data (profile image for avatar, personal information) capture work. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{vce}_{xdu} is the XR device captured avatar data upload delay to the digital worker device at the MEC server ($\beta^{vce}_{xdu} = \frac{\Psi^{vce}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{vce}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{vce}_{di} is the data size that consists of the XR device captured avatar creation data (profile image for avatar, personal information).

β^{vce}_{pvp} is the primary virtual network function processing delay at the digital work node server for the metaverse-based virtual clothing application ($\beta^{vce}_{pvp} = \frac{q^{vce}_{nfw}+q^{vce}_{nlb}+q^{vce}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{vce}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{vce}_{nfw}, q^{vce}_{nlb}, and q^{vce}_{npdi} are the virtual network function workload amounts for firewall, load balancing, and packet digital inspection for virtual clothing applications based offloaded data in the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node.

Ψ^{vce}_{tvpd} is the exchanged data amount (XR device to digital worker MEC server) during virtual network processing operation from one virtual network function processing server or node to another virtual network function processing node during the virtual clothing experience application.

β^{vce}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., avatar creation at metaverse, rendering frame, add static data, create environment) and result generation delay ($\beta^{vce}_{dwp} = \frac{q^{vce}_{dwp}}{\Lambda_{dd}}$). Where q^{vce}_{dwp} is the number of workloads for the offloaded data processing and result generation during virtual clothing application execution at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{vce}_{xpp} is the user XR device-based persons shopping data selection and cloth selection (for the avatar at metaverse) delay ($\beta^{vce}_{xpp} = \frac{q^{vce}_{xpp}}{\Lambda_{pd}}$). q^{vce}_{xpp} is the workload for XR device-based persons shopping data selection and cloth selection (for the avatar in the metaverse) work.

β^{vce}_{xduu} is the XR device captured shopping data upload delay to digital worker device at the MEC server ($\beta^{vce}_{xduu} = \frac{\Psi^{vce}_{dii}}{J_{wr}} * u_{wr} + \frac{\Psi^{vce}_{dii}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{vce}_{dii} is the data size that consists of the XR device captured shopping data (selected cloth for avatar).

β^{vce}_{pvpp} is the secondary virtual network function processing delay at the digital work node server during the second data offload operation for the metaverse-based virtual clothing application. ($\beta^{vce}_{pvpp} = \frac{q^{vce}_{nfw}+q^{vce}_{nlb}+q^{vce}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{vce}_{tvpd}}{J_{lr}} * u_{lr}$).

β^{vce}_{dwpp} is the digital work node (at the MEC server) based secondary offloaded data processing (i.e., visualize avatar with selected users cloth, haptic sensation generation, rendering frame) and result generation delay ($\beta^{vce}_{dwpp} = \frac{q^{vce}_{dwpp}}{\Lambda_{dd}}$). Where q^{vce}_{dwpp} is the number of workloads for the second offloaded data processing and result generation during virtual clothing application execution at the digital worker node.

β^{vce}_{rdt} is the digital work node processed virtual clothing experience resultant data dispatch delay to players XR device ($\beta^{vce}_{rdt} = \frac{\Psi^{vce}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{vce}_{rdt}}{J_{or}} * u_{or} + \beta^k_{pp} + \beta^k_{rwd}$). Where Ψ^{vce}_{rdt} is the output data size that consists of the virtual clothing experience applications resultant data amount.

β^{vce}_{svp} is the tertiary virtual network function processing delay at the digital work node before the virtual clothing experience application output data reception at the players XR device ($\beta^{vce}_{svp} = \frac{q^{vce}_{npdi}+q^{vce}_{nfw}+q^{vce}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{vce}_{stvd}}{J_{lr}} * u_{lr}$).

q^{vce}_{npdi}, q^{vce}_{nfw}, and q^{vce}_{nid} are the virtual network function workload amounts for DPI, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{vce}_{stvd} is the tertiary exchanged data amount during virtual network processing operation during metaverse-based virtual clothing application execution from one virtual network function processing server or node to another virtual network function processing node.

β^{vce}_{xpr} is the receiver user or XR device-based virtual clothing experience resultant data (rendered frame with immersion) visualization and haptic sensation reception delay by haptic jacket ($\beta^{vce}_{xpr} = \frac{q^{vce}_{xpr}}{\Lambda_{pd}}$). q^{vce}_{xpr} is the workload (virtual clothing experience applications output data visualization with haptic sensation) for the receiver device (XR device and haptic jacket). Λ_{pd} is the work processing power or CPU speed for the user's haptic jacket or XR device.

The fourth application that this paper is going to analyze is a metaverse-based healthcare application. The work completion delay for metaverse-based healthcare application β^k_{mh} execution is investigated by:

$$\begin{aligned}\beta^k_{mh} = \sum_{k=1}^{z} (&\beta^{mh}_{cim} + \beta^{mh}_{trd} + \beta^{mh}_{tmp} + \beta^{mh}_{sip} + \beta^{mh}_{xpu} + \beta^{mh}_{xdu} \\ &+ \beta^{mh}_{pvp} + \beta^{mh}_{dwp} + \beta^{mh}_{tpd} + \beta^{mh}_{pvpp} + \beta^{mh}_{xpuu} \\ &+ \beta^{mh}_{rdt} + \beta^{mh}_{svp} + \beta^{mh}_{xpr}) \quad (8)\end{aligned}$$

β^{mh}_{cim} is the initial readyness and metaverse connectivity delay for the user and XR device. β^{mh}_{trd} is the metaverse-based healthcare application that requests dispatch delay to a nearby task manager. β^{mh}_{tmp} is the digital task manager-based healthcare applications initial instruction generation delay for the workers. β^{mh}_{sip} is a metaverse-based healthcare application for work processing and instruction transfer delay from the digital task manager to workers. β^{mh}_{cim}, β^{mh}_{trd}, β^{mh}_{tmp}, and β^{mh}_{sip} are calculated by using the similar calculation formula as β^k_{cim}, β^k_{trd}, β^k_{tmp}, and β^k_{sip} (see XR-based gaming application discussion).

β^{mh}_{xpu} is the user XR device-based persons own health-care data (MRI data, CT-scan data, body temperature, blood rate data) capture delay ($\beta^{mh}_{xpu} = \frac{q^{mh}_{xpu}}{\Lambda_{pd}}$). q^{mh}_{xpu} is the workload for XR device persons own health-care data (MRI data, CT-scan data, body temperature, blood rate data) capture work. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{mh}_{xdu} is the XR device captured healthcare data upload delay to digital worker device at the MEC server ($\beta^{mh}_{xdu} = \frac{\Psi^{mh}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{mh}_{di}}{J_{or}} * u_{or} + \beta^k_{pp} + \beta^k_{rwd}$). Where Ψ^{mh}_{di} is the data size that consists of the XR device captured persons health-care information.

β^{mh}_{pvp} is the primary virtual network function processing delay at the digital work node server for the metaverse-based healthcare application ($\beta^{mh}_{pvp} = \frac{q^{mh}_{nfw}+q^{mh}_{npdi}+q^{mh}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{mh}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{mh}_{nfw}, q^{mh}_{npdi}, and q^{mh}_{nid} are the virtual network function workload amounts for firewalls, digital packet inspection,and intrusion detection for healthcare applications based on offloaded data on the MEC server, respectively. Λ_{dd} is the virtual network function processing power at the digital worker node.

Ψ^{mh}_{tvpd} is the exchanged data amount (XR device to digital worker MEC server) during virtual network processing operation from one virtual network function processing server or node to another virtual network function processing node during metaverse-based healthcare application execution.

β^{mh}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., predict disease and medicine prescription) and result generation delay ($\beta^{mh}_{dwp} = \frac{q^{mh}_{dwp}}{\Lambda_{dd}}$). Where q^{mh}_{dwp} is the number of workloads for the offloaded data processing and result generation during healthcare application execution at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{mh}_{tpd} is the digital worker processed data transfer delay to doctor device from MEC server ($\beta^{mh}_{tpd} = \frac{\Psi^{mh}_{dii}}{J_{wr}} * u_{wr} + \frac{\Psi^{mh}_{dii}}{J_{or}} * u_{or} + \beta^k_{pp} + \beta^k_{rwd}$). Where Ψ^{mh}_{dii} is the data size that consists of the digital worker's (MEC) processed output data for healthcare services.

β^{mh}_{pvpp} is the secondary virtual network function processing delay at the virtual network function processing server for the metaverse-based healthcare application ($\beta^{mh}_{pvpp} = \frac{q^{mh}_{nfw}+q^{mh}_{npdi}+q^{mh}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{mh}_{tvpdd}}{J_{lr}} * u_{lr}$).

q^{mh}_{nfw}, q^{mh}_{npdi}, and q^{mh}_{nid} are the virtual network function workload amounts for firewalls, digital packet inspection,and intrusion detection for healthcare applications based on offloaded data on the MEC server, respectively. Λ_{dd} is the virtual network function processing power at the virtual network function processing server.

Ψ^{mh}_{tvpdd} is the exchanged data amount from one virtual network function processing server or node to another virtual network function processing node during secondary VNF processing.

β^{mh}_{xpuu} is the doctors XR device-based disease and medication result selection delay ($\beta^{mh}_{xpuu} = \frac{q^{mh}_{xpuu}}{\Lambda_{pd}}$). q^{mh}_{xpuu} is the workload for doctors XR device-based disease and medication result generation.

β^{mh}_{rdt} is the doctor device processed resultant healthcare data dispatch delay to users or patients XR device ($\beta^{mh}_{rdt} = \frac{\Psi^{mh}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{mh}_{rdt}}{J_{or}} * u_{or} + \beta^k_{pp} + \beta^k_{rwd}$). Where Ψ^{mh}_{rdt} is the output data size that consists of the resultant healthcare data amount.

β^{mh}_{svp} is the tertiary virtual network function processing delay at the virtual network function processing node before the output data reception at the players XR device ($\beta^{mh}_{svp} = \frac{q^{mh}_{atn}+q^{mh}_{nfw}+q^{mh}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{mh}_{stvd}}{J_{lr}} * u_{lr}$).

q^{mh}_{atn}, q^{mh}_{nfw}, and q^{mh}_{nid} are the virtual network function workload amounts for NAT, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{mh}_{stvd} is the tertiary exchanged data amount during a virtual network processing operation from one virtual network function processing server or node to another virtual network function processing node.

β^{mh}_{xpr} is the receiver user device or XR device-based healthcare resultant data visualization delay ($\beta^{mh}_{xpr} = \frac{q^{mh}_{xpr}}{\Lambda_{pd}}$). q^{mh}_{xpr} is the workload for the receiver device (XR device). Λ_{pd} is the work processing power or CPU speed for the user XR device.

The fifth application considered in this paper is socialization activities in the metaverse via the avatar. The work completion delay for socialization activities in the metaverse via the avatar β^{k}_{sa} is given by:

$$\begin{aligned}\beta^{k}_{sa} = \sum_{k=1}^{z}(&\beta^{sa}_{cim} + \beta^{sa}_{trd} + \beta^{sa}_{tmp} + \beta^{sa}_{sip} + \beta^{sa}_{xpu} + \beta^{sa}_{xdu} \\ &+ \beta^{sa}_{pvp} + \beta^{sa}_{dwp} + \beta^{sa}_{xpp} + \beta^{sa}_{xduu} + \beta^{sa}_{pvpp} + \beta^{sa}_{dwpp} \\ &+ \beta^{sa}_{rdt} + \beta^{sa}_{svp} + \beta^{sa}_{xpr}) \qquad (9)\end{aligned}$$

β^{sa}_{cim} is the initial device readyness and internet connectivity delay for the user and XR device. β^{sa}_{trd} is the socialization activities (in the metaverse) application request dispatch delay to nearby task manager. β^{sa}_{tmp} is the digital task manager-based socialization activity applications initial instruction generation delay for the workers. β^{sa}_{sip} is a metaverse-based socialization application for work processing and instruction transfer delay from the digital task manager to workers. β^{sa}_{cim}, β^{sa}_{trd}, β^{sa}_{tmp}, and β^{sa}_{sip} are calculated by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see XR-based gaming application discussion).

β^{sa}_{xpu} is the user XR device-based avatar creation preliminary data capture delay ($\beta^{sa}_{xpu} = \frac{q^{sa}_{xpu}}{\Lambda_{pd}}$). q^{sa}_{xpu} is the workload for XR device-based person data (profile image for avatar, pose, background data, personal information) capture work for avatar creation in the metaverse. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{sa}_{xdu} is the XR device captured avatar creation data upload delay to digital worker device at the MEC server ($\beta^{sa}_{xdu} = \frac{\Psi^{sa}_{di}}{J_{wr}}*u_{wr}+\frac{\Psi^{sa}_{di}}{J_{or}}*u_{or}+\beta^{k}_{pp}+\beta^{k}_{rwd}$). Where Ψ^{sa}_{di} is the data size that consists of the XR device captured avatar creation data (profile image for avatar, pose, background data, personal information).

β^{sa}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the metaverse-based socialization application ($\beta^{sa}_{pvp} = \frac{q^{sa}_{nfw}+q^{sa}_{nlb}+q^{sa}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{sa}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{sa}_{nfw}, q^{sa}_{nlb}, and q^{sa}_{nid} are the virtual network function workload amounts for firewall, load balancing, and IDS for metaverse socialization applications during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node.

Ψ^{sa}_{tvpd} is the exchanged data amount during a virtual network processing operation from one virtual network function processing server or node to another virtual network function processing node during a socialization application.

β^{sa}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., avatar creation at metaverse, rendering frame, add static data, create environment) and result generation delay ($\beta^{sa}_{dwp} = \frac{q^{sa}_{dwp}}{\Lambda_{dd}}$). Where q^{sa}_{dwp} is the number of workloads for the offloaded data processing and result generation during meaverse socialization application execution at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{sa}_{xpp} is the user XR device-based avatar socialization data selection (for one avatar to another avatar interaction in the metaverse) delay ($\beta^{sa}_{xpp} = \frac{q^{sa}_{xpp}}{\Lambda_{pd}}$). q^{sa}_{xpp} is the workload for XR device-based avatar socialization data selection work.

β^{sa}_{xduu} is the XR device captured socialization data upload delay to digital worker device at the MEC server ($\beta^{sa}_{xduu} = \frac{\Psi^{sa}_{dii}}{J_{wr}} * u_{wr} + \frac{\Psi^{sa}_{dii}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{sa}_{dii} is the data size that consists of the XR device captured socialization or interaction data (selected for avatar).

β^{sa}_{pvpp} is the secondary virtual network function processing delay at the digital work node server during secondary interaction data offload operation for the metaverse-based socialization application. ($\beta^{sa}_{pvpp} = \frac{q^{sa}_{nfw}+q^{sa}_{nid}+q^{sa}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{sa}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{sa}_{nfw}, q^{sa}_{nid}, and q^{sa}_{npdi} are workloads for firewall, intrusion detection,and digital packet inspection operations at the virtual network function processing device during secondary offload operations.

β^{sa}_{dwpp} is the digital work node (at the MEC server) based secondary offloaded data processing (i.e., visualize action, avatar to avatar interaction, or social conversion) and result generation delay ($\beta^{sa}_{dwpp} = \frac{q^{sa}_{dwpp}}{\Lambda_{dd}}$). Where q^{sa}_{dwpp} is the number of workloads for secondary offloaded data processing and result generation during metaverse socialization execution at the digital worker node.

β^{sa}_{rdt} is the digital work node processed metaverse avatar socialization resultant data dispatch delay to players XR device ($\beta^{sa}_{rdt} = \frac{\Psi^{sa}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{sa}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{sa}_{rdt} is the output data size that consists of the avatar-to-avatar interaction experience application resultant data amount.

β^{sa}_{svp} is the tertiary virtual network function processing delay at the digital work node before the metaverse avatar socialization resultant output data reception at the players XR device ($\beta^{sa}_{svp} = \frac{q^{sa}_{npdi}+q^{sa}_{nfw}+q^{sa}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{sa}_{stvd}}{J_{lr}} * u_{lr}$).

q^{sa}_{npdi}, q^{sa}_{nfw}, and q^{sa}_{nid} are the virtual network function workload amounts for DPI, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{sa}_{stvd} is the tertiary exchanged data amount during virtual network processing operation during metaverse-based avatar interaction application execution from one virtual network function processing server or node to another virtual network function processing node.

β^{sa}_{xpr} is the receiver user or XR device-based metaverse avatar socialization result visualization and haptic sensation reception delay by haptic jacket ($\beta^{sa}_{xpr} = \frac{q^{sa}_{xpr}}{\Lambda_{pd}}$). q^{sa}_{xpr} is the workload for the receiver device (XR device and haptic jacket) for the avatar socialization application. Λ_{pd} is the work processing power or CPU speed for the user's haptic jacket or XR device.

The sixth application considered in this paper is a hologram-based remote presence application for video conferencing. The work completion delay for the hologram-based remote presence application β^{k}_{hrp} is given by:

$$\beta^{k}_{hrp} = \sum_{k=1}^{z}(\beta^{hrp}_{cim} + \beta^{hrp}_{trd} + \beta^{hrp}_{tmp} + \beta^{hrp}_{sip} + \beta^{hrp}_{xpu} + \beta^{hrp}_{xdu} + \beta^{hrp}_{pvp} + \beta^{hrp}_{dwp} + +\beta^{hrp}_{rdt} + \beta^{hrp}_{svp} + \beta^{hrp}_{xpuu} + \beta^{hrp}_{xpr}) \quad (10)$$

β^{hrp}_{cim} is the initial device readyness and internet connectivity delay for the user and XR device. β^{hrp}_{trd} is the hologram-based remote presence application that requests dispatch delay to the nearby task manager. β^{hrp}_{tmp} is the digital task manager-based hologram-based remote presence applications instruction generation delay for the workers. β^{hrp}_{sip} is a hologram-based remote presence application that works with instruction transfer delays from the digital task manager to workers. β^{hrp}_{cim}, β^{hrp}_{trd}, β^{hrp}_{tmp}, and β^{hrp}_{sip} are measured by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{hrp}_{xpu} is the user XR device-based voice and video data capture delay for remote telepresence applications ($\beta^{hrp}_{xpu} = \frac{q^{hrp}_{xpu}}{\Lambda_{pd}}$). q^{hrp}_{xpu} is the workload for XR device-based voice and video data capture work. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{hrp}_{xdu} is the XR device captured data upload delay to the digital worker device at the MEC server ($\beta^{hrp}_{xdu} = \frac{\Psi^{hrp}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{hrp}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{hrp}_{di} is the data size that consists of the XR device captured voice and video data amounts for remote telepresence applications.

β^{hrp}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the metaverse-based remote telepresence application ($\beta^{hrp}_{pvp} = \frac{q^{hrp}_{nfw}+q^{hrp}_{atn}+q^{hrp}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{hrp}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{hrp}_{nfw}, q^{hrp}_{atn}, and q^{hrp}_{nid} are the virtual network function workload amounts for firewall, NAT, and IDS for remote telepresence applications during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{hrp}_{tvpd} is the exchanged data amount during virtual network processing operation execution.

β^{hrp}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., compress, encoding, rendering, hologram generation) and result generation delay ($\beta^{hrp}_{dwp} = \frac{q^{hrp}_{dwp}}{\Lambda_{dd}}$). Where q^{hrp}_{dwp} is the number of workloads for the offloaded data processing and result generation during hologram-based telepresence application execution at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{hrp}_{rdt} is the digital work node processed hologram-based telepresence resultant data dispatch delay to players XR device ($\beta^{hrp}_{rdt} = \frac{\Psi^{hrp}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{hrp}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{hrp}_{rdt} is the output data size that consists of the hologram-based telepresence resultant data amount.

β^{hrp}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the hologram-based telepresence resultant data reception at the players XR device ($\beta^{hrp}_{svp} = \frac{q^{hrp}_{atn}+q^{hrp}_{nfw}+q^{hrp}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{hrp}_{stvd}}{J_{lr}} * u_{lr}$).

q^{hrp}_{atn}, q^{hrp}_{nfw}, and q^{hrp}_{nid} are the virtual network function workload amounts for NAT, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{hrp}_{stvd} is the secondary exchanged data amount during virtual network processing operation during a hologram-based telepresence application.

β^{hrp}_{xpuu} is the receiver user or XR device-based final result generation delay by the user XR device ($\beta^{hrp}_{xpuu} = \frac{q^{hrp}_{xpuu}}{\Lambda_{pd}}$). q^{hrp}_{xpuu} is the workload for the receiver device (XR device) for decompressing, decoding, and final hologram-based resultant data generation. Λ_{pd} is the work processing power or CPU speed for the user or XR device.

β^{hrp}_{xpr} is the receiver user or XR device hologram-based remote tele-presence result visualization and execution delay ($\beta^{hrp}_{xpr} = \frac{q^{hrp}_{xpr}}{\Lambda_{pd}}$). q^{hrp}_{xpr} is the workload for the receiver device (XR device and haptic device) for final hologram-based remote telepresence data visualization.

Our next considered metaverse application is a federated learning (FL)-based surveillance application. The work completion delay for the federated learning-based surveillance application β^{k}_{fs} is calculated by:

$$\beta^{k}_{fs} = \sum_{k=1}^{z}(\beta^{fs}_{cim} + \beta^{fs}_{trd} + \beta^{fs}_{tmp} + \beta^{fs}_{sip} + \beta^{fs}_{rdu} + \beta^{fs}_{xpu} + \beta^{fs}_{xdu} + \beta^{fs}_{pvp} + \beta^{fs}_{dwp} + \beta^{fs}_{dwpp} + +\beta^{fs}_{rdt} + \beta^{fs}_{svp} + \beta^{fs}_{xpuu}) \quad (11)$$

β^{fs}_{cim} is the initial device readyness and internet connectivity delay for the user and XR device. β^{fs}_{trd} is the federated learning (FL)-based surveillance application that requests dispatch delay to the nearby task manager. β^{fs}_{tmp} is the digital task manager-based federated learning (FL) surveillance application instruction generation delay for the workers. β^{fs}_{sip} is FL applications work instruction transfer delay from the digital task manager to workers. β^{fs}_{cim}, β^{fs}_{trd}, β^{fs}_{tmp}, and β^{fs}_{sip} are measured by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{fs}_{rdu} is the XR device-based global deep learning model (i.e., CNN for human recognition) for data download delay from a digital worker device or the MEC server $\beta^{fs}_{rdu} = \frac{\Psi^{fs}_{dii}}{J_{wr}} * u_{wr} + \frac{\Psi^{fs}_{dii}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{fs}_{dii} is the data size that consists of the global deep learning model (i.e., CNN for human recognition) data for FL-based human surveillance applications.

β^{fs}_{xpu} is the user XR device-based video data capture, local model training, and updated parameter generation delay ($\beta^{fs}_{xpu} = \frac{q^{fs}_{xpu}}{\Lambda_{pd}}$). q^{fs}_{xpu} is the workload for XR device-based data capture and local training work. Λ_{pd} is the work processing power or CPU speed for the XR device.

β^{fs}_{xdu} is the XR device generated local model parameter data upload delay to the digital worker device at the MEC server ($\beta^{fs}_{xdu} = \frac{\Psi^{fs}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{fs}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{fs}_{di} is the uploaded data size that consists of the XR device-generated local model parameter for FL-based human surveillance applications.

β^{fs}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the FL-based surveillance application ($\beta^{fs}_{pvp} = \frac{q^{fs}_{nfw}+q^{fs}_{atn}+q^{fs}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{fs}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{fs}_{nfw}, q^{fs}_{atn}, and q^{fs}_{nid} are the virtual network function workload amounts for firewall, NAT, and IDS during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{fs}_{tvpd} is the exchanged data amount during virtual network processing operation execution.

β^{fs}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (local model data aggregation and global model update) and result generation delay ($\beta^{fs}_{dwp} = \frac{q^{fs}_{dwp}}{\Lambda_{dd}}$). Where q^{fs}_{dwp} is the number of workloads for the offloaded data processing and result generation during Fl-based surveillance application execution at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{fs}_{dwpp} is the digital work node (at the MEC server) based secondary data processing (integrate learning result to the avatar) delay ($\beta^{fs}_{dwpp} = \frac{q^{fs}_{dwp}}{\Lambda_{dd}}$). Where q^{fs}_{dwpp} is the number of workloads for the secondary operation (avatar learning) processing at the digital worker node.

β^{fs}_{rdt} is the digital work node processed resultant global FL model data dispatch delay to players XR device ($\beta^{fs}_{rdt} = \frac{\Psi^{fs}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{fs}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{fs}_{rdt} is the output data size that consists of the global FL model data amount.

β^{fs}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the players XR device ($\beta^{fs}_{svp} = \frac{q^{fs}_{atn}+q^{fs}_{nfw}+q^{fs}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{fs}_{stvd}}{J_{lr}} * u_{lr}$).

q^{fs}_{atn}, q^{fs}_{nfw}, and q^{fs}_{nid} are the virtual network function workload amounts for NAT, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{fs}_{stvd} is the secondary exchanged data amount during virtual network processing operations during FL-based surveillance applications.

β^{fs}_{xpuu} is the receiver user or XR device-based final result generation delay by the user XR device ($\beta^{fs}_{xpuu} = \frac{q^{fs}_{xpuu}}{\Lambda_{pd}}$). q^{fs}_{xpuu} is the workload for the receiver device (XR device). Λ_{pd} is the work processing power or CPU speed for the user or XR device.

Our next considered application for analysis is blockchain (BC) and NFT-based visual art selling in the metaverse. The work completion delay for blockchain and NFT-based visual art selling in the metaverse β^{k}_{bnas} is given by:

$$\begin{aligned}\beta^{k}_{bnas} = \sum_{k=1}^{z}(&\beta^{bnas}_{cim} + \beta^{bnas}_{trd} + \beta^{bnas}_{tmp} + \beta^{bnas}_{sip} + \\ &\beta^{bnas}_{bcr} + \beta^{bnas}_{xpu} + \beta^{bnas}_{xdu} + \beta^{bnas}_{pvp} + \beta^{bnas}_{dwp} + \\ &\beta^{bnas}_{xpuu} + \beta^{bnas}_{xduu} + \beta^{bnas}_{dwpp} + \beta^{bnas}_{bcc} + \beta^{bnas}_{btt} \\ &+ \beta^{bnas}_{bvm} + \beta^{bnas}_{dwps} + \beta^{bnas}_{rdt} + \beta^{bnas}_{svp} + \beta^{bnas}_{xpr}) \quad (12)\end{aligned}$$

β^{bnas}_{cim} is the initial device readyness and internet connectivity delay for the user and XR device. β^{bnas}_{trd} is the blockchain (BC) and NFT-based visual art selling request dispatch delay to the nearby task manager. β^{bnas}_{tmp} is the digital task manager-based applications instruction generation delay for the workers. β^{bnas}_{sip} is a BC-based NFT trading application that works with instruction transfer delays from the digital task manager to workers. β^{bnas}_{cim}, β^{bnas}_{trd}, β^{bnas}_{tmp}, and β^{bnas}_{sip} are analyzed by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{bnas}_{bcr} is the user XR device and MEC based initial blockchain operation delay ($\beta^{bnas}_{bcr} = \frac{q^{bnas}_{bcr}}{\Lambda_{pd}} + \frac{q^{bnas}_{bcrr}}{\Lambda_{dd}} + \beta^{bnas}_{bcdt}$). q^{bnas}_{bcr} is the workload for the XR device during the initial blockchain operation (BC registration, smart contract creation, key transfer, user verification). q^{bnas}_{bcrr} is the workload for the edge blockchain device during the initial blockchain operation (BC registration, smart contract creation, key transfer, user verification). β^{bnas}_{bcdt} is the data transfer delay between the user XR device and the edge blockchain device during the initial blockchain operation. Λ_{pd} is the work processing power or CPU speed for the XR device. Λ_{dd} is the work processing power or CPU speed for the edge blockchain device.

β^{bnas}_{xpu} is the user XR device-based avatar data capture and art selling data capture delay ($\beta^{bnas}_{xpu} = \frac{q^{bnas}_{xpu}}{\Lambda_{pd}}$). q^{bnas}_{xpu} is the workload for XR device-based avatar creation and art-selling data capture work. β^{bnas}_{xdu} is the XR device captured data upload delay to the digital worker device at the MEC server ($\beta^{bnas}_{xdu} = \frac{\Psi^{bnas}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{bnas}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{bnas}_{di} is the data size that consists of the XR device captured avatar creation and visual art selling post data.

β^{bnas}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the BC and NFT-based trading applications ($\beta^{bnas}_{pvp} = \frac{q^{bnas}_{nfw}+q^{bnas}_{atn}+q^{bnas}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{bnas}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{bnas}_{nfw}, q^{bnas}_{atn}, and q^{bnas}_{nid} are the virtual network function workload amounts for firewall, NAT, and IDS for NFT-based art trading applications during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{bnas}_{tvpd} is the exchanged data amount during virtual network processing operation execution.

β^{bnas}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., digital art selling post creation by avatar at the metaverse platform) and result generation delay ($\beta^{bnas}_{dwp} = \frac{q^{bnas}_{dwp}}{\Lambda_{dd}}$). Where q^{bnas}_{dwp} is the number of workloads for the offloaded data processing and result generation during NFT-based art selling at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{bnas}_{xpuu} is the buyer user or XR device-based visual art buying information generation delay ($\beta^{bnas}_{xpuu} = \frac{q^{bnas}_{xpuu}}{\Lambda_{pd}}$). q^{bnas}_{xpuu} is the workload for the buyer user or XR device-based visual art buying information generation work.

β^{bnas}_{xduu} is the buyer XR device captured data upload delay to digital worker device at the MEC server ($\beta^{bnas}_{xduu} = \frac{\Psi^{bnas}_{dii}}{J_{wr}} * u_{wr} + \frac{\Psi^{bnas}_{dii}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{bnas}_{dii} is the data size that consists of the buyer's XR device captured visual art buying data via the exchange of cryptocurrency.

β^{bnas}_{dwpp} is the digital work node (at the MEC server) based offloaded data processing (i.e., digital art selling confirmation by seller avatar and NFT transfer to buyer account, cryptocurrency transfer to seller account) and result generation delay ($\beta^{bnas}_{dwpp} = \frac{q^{bnas}_{dwpp}}{\Lambda_{dd}}$). Where q^{bnas}_{dwpp} is the number of secondary workloads for the offloaded data processing and result generation during NFT-based art selling at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{bnas}_{bcc} is the digital work node (blockchain master node at the MEC server) based block creation and hashing delay ($\beta^{bnas}_{bcc} = \frac{q^{bnas}_{bcc}}{\Lambda_{dd}}$). Where q^{bnas}_{bcc} is the number of workloads for block creation at the digital worker node (blockchain node at MEC). Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{bnas}_{btt} is the block data dispatch delay to verifier devices (MEC server) at the blockchain network ($\beta^{bnas}_{btt} = \frac{\Psi^{bnas}_{dib}}{J_{wr}} * u_{wr} + \frac{\Psi^{bnas}_{dib}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{bnas}_{dib} is the data size that consists of the block creation data.

β^{bnas}_{bvm} is the digital work node (blockchain verifier node at the MEC server) based block verification delay ($\beta^{bnas}_{bvm} = \frac{q^{bnas}_{bvm}}{\Lambda_{dd}}$). Where q^{bnas}_{bvm} is the number of workloads for the block verification at the digital worker node (blockchain verifier node at MEC). Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{bnas}_{dwps} is the digital work node (blockchain master node at the MEC server) based blockchain ledger update delay after verification ($\beta^{bnas}_{dwps} = \frac{q^{bnas}_{dwps}}{\Lambda_{dd}}$). Where q^{bnas}_{dwps} is the number of workloads for the ledger update at the digital worker node (blockchain node at MEC). Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{bnas}_{rdt} is the digital work node processed resultant NFT trading updated data dispatch delay to users XR device ($\beta^{bnas}_{rdt} = \frac{\Psi^{bnas}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{bnas}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{bnas}_{rdt} is the output data size that consists of the resultant NFT trading finalization data.

β^{bnas}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the users XR device ($\beta^{bnas}_{svp} = \frac{q^{bnas}_{atn}+q^{bnas}_{nfw}+q^{bnas}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{bnas}_{stvd}}{J_{lr}} * u_{lr}$).

q^{bnas}_{atn}, q^{bnas}_{nfw}, and q^{bnas}_{nid} are the virtual network function workload amounts for NAT, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{bnas}_{stvd} is the secondary exchanged data amount during the virtual network processing operation during the NFT trading application.

β^{bnas}_{xpr} is the receiver user or XR device-based final NFT trading result display delay ($\beta^{bnas}_{xpr} = \frac{q^{bnas}_{xpr}}{\Lambda_{pd}}$). q^{bnas}_{xpr} is the workload for the receiver device (XR device) during resultant data visualization.

The next considered application is digital twin-based robot selection for manufacturing work. The work completion delay for the digital twin-based simulation application β^{k}_{drs} is given by:

$$\beta^{k}_{drs} = \sum_{k=1}^{z}(\beta^{drs}_{cim} + \beta^{drs}_{trd} + \beta^{drs}_{tmp} + \beta^{drs}_{sip} + \beta^{drs}_{xpu} + \beta^{drs}_{xdu} + \beta^{drs}_{pvp} + \beta^{drs}_{dwp} + \beta^{drs}_{xpuu} + \beta^{drs}_{xduu} + \beta^{drs}_{dwop} + \beta^{drs}_{rdt} + \beta^{drs}_{dwpp} + \beta^{drs}_{rdtt} + \beta^{drs}_{svp} + \beta^{drs}_{xpr}) \quad (13)$$

β^{drs}_{cim} is the initial device readyness and internet connectivity delay for the user and XR device. β^{drs}_{trd} is the digital twin-based simulation (for best robot selection) application that requests dispatch delay to the nearby task manager. β^{drs}_{tmp} is the digital task manager-based digital twin applications instruction generation delay for the workers. β^{drs}_{sip} is the digital twin applications work instruction transfer delay from the digital task manager to workers. β^{drs}_{cim}, β^{drs}_{trd}, β^{drs}_{tmp}, and β^{drs}_{sip} are measured by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{drs}_{xpu} is the user device or robot-based data capture operation for digital twin creation ($\beta^{drs}_{xpu} = \frac{q^{drs}_{xpu}}{\Lambda_{pd}}$). q^{drs}_{xpu} is the workload for the user device or robot-based data capture operation for digital twin creation. Λ_{pd} is the work processing power or CPU speed for the user device.

β^{drs}_{xdu} is the user device or robot captured data upload delay to the digital worker device at the MEC server for digital twin creation ($\beta^{drs}_{xdu} = \frac{\Psi^{drs}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{drs}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{drs}_{di} is the data size that consists of the user device or robot device captured data amount for the digital twin-based robot selection application.

β^{drs}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the digital twin simulation application ($\beta^{drs}_{pvp} = \frac{q^{drs}_{nfw}+q^{drs}_{atn}+q^{drs}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{drs}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{drs}_{nfw}, q^{drs}_{atn}, and q^{drs}_{nid} are the virtual network function workload amounts for firewall, NAT, and IDS for the digital twin simulation application during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{drs}_{tvpd} is the exchanged data amount during virtual network processing operation execution.

β^{drs}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., digital twin creation and synchroization) delay ($\beta^{drs}_{dwp} = \frac{q^{drs}_{dwp}}{\Lambda_{dd}}$). Where q^{drs}_{dwp} is the number of workloads for the offloaded data processing at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{drs}_{xpuu} is the robot present state data capture operation for digital twin creation ($\beta^{drs}_{xpuu} = \frac{q^{drs}_{xpuu}}{\Lambda_{pd}}$). q^{drs}_{xpuu} is the workload for the robot's own data capture operation for digital twin simulation.

β^{drs}_{xduu} is the robot's own data upload delay to the digital worker device at the MEC server for digital twin-based simulation operation ($\beta^{drs}_{xduu} = \frac{\Psi^{drs}_{dii}}{J_{wr}} * u_{wr} + \frac{\Psi^{drs}_{dii}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{drs}_{dii} is the data size that consists of the robot device's own data amount for the digital twin simulation-based robot selection application.

β^{drs}_{dwop} is the digital twin node (at the MEC server) based offloaded data processing (i.e., simulate the behavior of robots for manufacturing jobs, robot status prediction during work execution, fuel status prediction, and fault detection) delay ($\beta^{drs}_{dwop} = \frac{q^{drs}_{dwop}}{\Lambda_{dd}}$). Where q^{drs}_{dwop} is the number of workloads for the offloaded data processing at the digital twin node. Λ_{dd} is the workload porcessing power at the digital twin node.

β^{drs}_{rdt} is the digital twin-generated resultant data dispatch delay to the MEC device ($\beta^{drs}_{rdt} = \frac{\Psi^{drs}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{drs}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{drs}_{rdt} is the output data size that consists of the digital twin simulated data amount.

β^{drs}_{dwpp} is the digital worker node (at the MEC server) based simulated data processing (i.e., best robot selection for manufacturing job from digital twin simulated data) delay ($\beta^{drs}_{dwpp} = \frac{q^{drs}_{dwpp}}{\Lambda_{dd}}$). Where q^{drs}_{dwpp} is the number of workloads for the offloaded data processing at the digital worker node. Λ_{dd} is the workload-consuming power at the digital worker or MEC node.

β^{drs}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the robot selection or resultant data reception at the users device ($\beta^{drs}_{svp} = \frac{q^{drs}_{atn}+q^{drs}_{nfw}+q^{drs}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{drs}_{stvd}}{J_{lr}} * u_{lr}$).

q^{drs}_{atn}, q^{drs}_{nfw}, and q^{drs}_{nid} are the virtual network function workload amounts for NAT, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{drs}_{stvd} is the secondary exchanged data amount during virtual network processing operation during a digital twin-based application.

β^{drs}_{xpr} is the receiver user or XR device-based remote robot selection result visualization delay ($\beta^{drs}_{xpr} = \frac{q^{drs}_{xpr}}{\Lambda_{pd}}$). q^{drs}_{xpr} is the workload for the receiver device (XR device and user device) for final robot selection result visualization.

Next, this article will provide an analytical model for non-metaverse-based industrial automation application execution. The work completion delay for industrial automation application execution β^{k}_{ia} is given by:

$$\begin{aligned}\beta^{k}_{ia} = \sum_{k=1}^{z}(&\beta^{ia}_{cim} + \beta^{ia}_{trd} + \beta^{ia}_{tmp} + \beta^{ia}_{sip} + \beta^{ia}_{xpu} + \beta^{ia}_{xdu} \\ &+ \beta^{ia}_{pvp} + \beta^{ia}_{dwp} + \beta^{ia}_{rdtc} + \beta^{ia}_{xpuc} + \beta^{ia}_{xpus} + \beta^{ia}_{rdth} \\ &+ \beta^{ia}_{xpuh} + \beta^{ia}_{rdtt} + \beta^{ia}_{svp} + \beta^{ia}_{xpr}) \quad (14)\end{aligned}$$

β^{ia}_{cim} is the initial device readyness and internet connectivity delay for the user, cobot, human device, video camera, and XR device. β^{ia}_{trd} is the industrial automation application (checking the shoe sole glueing process) execution request dispatch delay to nearby task manager. β^{ia}_{tmp} is the digital task manager-based industrial automation application instruction generation delay for the workers (cobot, human worker, video camera). β^{ia}_{sip} is an industrial automation application processing work instruction transfer delay from the digital task manager to workers. β^{ia}_{cim}, β^{ia}_{trd}, β^{ia}_{tmp}, and β^{ia}_{sip} are calculated by using the same calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{ia}_{xpu} is the video camera-based lether goods or shoe soles glue spraying image capture operation delay ($\beta^{ia}_{xpu} = \frac{q^{ia}_{xpu}}{\Lambda_{pd}}$). q^{ia}_{xpu} is the workload for video camera-based shoe sole image capture operations. Λ_{pd} is the work processing power or CPU speed for the video camera device.

β^{ia}_{xdu} is the users video camera captured shoe soles data upload delay to digital worker device at the MEC server ($\beta^{ia}_{xdu} = \frac{\Psi^{ia}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{ia}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{ia}_{di} is the data size that consists of the users video camera device captured data amount for shoe soles glue spray monitoring application.

β^{ia}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the industrial automation application ($\beta^{ia}_{pvp} = \frac{q^{ia}_{nfw}+q^{ia}_{atn}+q^{ia}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{ia}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{ia}_{nfw}, q^{ia}_{atn}, and q^{ia}_{nid} are the virtual network function workload amounts for firewall, NAT, and IDS for industrial automation applications during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{ia}_{tvpd} is the exchanged data amount during virtual network processing operation execution.

β^{ia}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., glue spraying path and point coordinate computation from sole image) delay ($\beta^{ia}_{dwp} = \frac{q^{ia}_{dwp}}{\Lambda_{dd}}$). Where q^{ia}_{dwp} is the number of workload for the offloaded data processing at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{ia}_{rdtc} is the digital work node generated resultant data dispatch delay to cobot (collaborative robot) device ($\beta^{ia}_{rdtc} = \frac{\Psi^{ia}_{rdtc}}{J_{wr}} * u_{wr} + \frac{\Psi^{ia}_{rdtc}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{ia}_{rdtc} is the output data size that consists of the digital work node-generated resultant data.

β^{ia}_{xpuc} is the cobot-based glue spraying operation completion delay based on MEC instruction ($\beta^{ia}_{xpuc} = \frac{q^{ia}_{xpuc}}{\Lambda_{pd}}$). q^{ia}_{xpuc} is the workload for cobot-based glue sparaying operations for industrial automation operations.

β^{ia}_{xpus} is the video camera-based glue spraying operation image capture delay based on MEC instruction after cobot work completion ($\beta^{ia}_{xpus} = \frac{q^{ia}_{xpus}}{\Lambda_{pd}}$). q^{ia}_{xpus} is the workload for video camera-based glue sparaying image capture for industrial automation operations.

β^{ia}_{rdth} is the video camera-based resultant data dispatch delay to human worker (mobile phone) device ($\beta^{ia}_{rdth} = \frac{\Psi^{ia}_{rdth}}{J_{wr}} * u_{wr} + \frac{\Psi^{ia}_{rdth}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{ia}_{rdth} is the video camera captured and offloaded resultant data size.

β^{ia}_{xpuh} is the human worker device based glue spraying operation checking or inspection delay based on MEC instruction after cobot and video camera work completion ($\beta^{ia}_{xpuh} = \frac{q^{ia}_{xpuh}}{\Lambda_{pd}}$). q^{ia}_{xpuh} is the workload for human worker device-based glue sparaying operation checking for industrial automation applications.

β^{ia}_{rdtt} is the human worker device based final sole glueing work inspection result or final output data dispatch delay to all worker and user devices ($\beta^{ia}_{rdtt} = \frac{\Psi^{ia}_{rdtt}}{J_{wr}} * u_{wr} + \frac{\Psi^{ia}_{rdtt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{ia}_{rdtt} is the human worker device-based offloaded resultant data size.

β^{ia}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the users device ($\beta_{svp}^{ia} = \frac{q_{npdi}^{ia}+q_{nfw}^{ia}+q_{nid}^{ia}}{\Lambda_{dd}} + \frac{\Psi_{stvd}^{ia}}{J_{lr}} * u_{lr}$).

q_{npdi}^{ia}, q_{nfw}^{ia}, and q_{nid}^{ia} are the virtual network function workload amounts for DPI, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ_{stvd}^{ia} is the secondary exchanged network function processing data amount during virtual network processing operation during industrial automation applications.

β_{xpr}^{ia} is the receiver user or XR device-based industrial automation application completion (sole glue spraying work) result visualization delay ($\beta_{xpr}^{ia} = \frac{q_{xpr}^{ia}}{\Lambda_{pd}}$). q_{xpr}^{ia} is the workload for the receiver device (XR device and user device) for final work completion result visualization.

Now, we will discuss the non-metaverse-based autonomous vehicle application execution delay. The work completion delay for autonomous vehicle application execution β_{av}^{k} is given by:

$$\beta_{av}^{k} = \sum_{k=1}^{z}(\beta_{cim}^{av} + \beta_{trd}^{av} + \beta_{tmp}^{av} + \beta_{sip}^{av} + \beta_{xpu}^{av} + \beta_{xdu}^{av} + \beta_{pvp}^{av} + \beta_{dwp}^{av} + \beta_{rdt}^{av} + \beta_{svp}^{av} + \beta_{xpr}^{av}) \quad (15)$$

β_{cim}^{av} is the initial device readyness and internet connectivity delay for the user, vehicle sensors, video camera, and XR device. β_{trd}^{av} is the industrial autonomous vehicle application (checking the traffic light or object detection in the road) execution request dispatch delay to nearby task manager. β_{tmp}^{av} is the digital task manager-based autonomous vehicle application instruction generation delay for the workers (MEC, video camera, vehicle steering). β_{sip}^{av} is an autonomous vehicle application processing work instruction transfer delay from the digital task manager to workers. β_{cim}^{av}, β_{trd}^{av}, β_{tmp}^{av}, and β_{sip}^{av} are calculated by using the similar calculation formula as β_{cim}^{k}, β_{trd}^{k}, β_{tmp}^{k}, and β_{sip}^{k} (see first XR based gaming application discussion).

β_{xpu}^{av} is the vehicle's video camera-based image capture (i.e., traffic light or road object for detection) operation delay ($\beta_{xpu}^{av} = \frac{q_{xpu}^{av}}{\Lambda_{pd}}$). q_{xpu}^{av} is the workload for video camera-based image capture (i.e., traffic light or road object detection) operations. Λ_{pd} is the work processing power or CPU speed for the vehicle's video camera device.

β_{xdu}^{av} is the vehicle video camera captured data upload delay to the digital worker device at the MEC server ($\beta_{xdu}^{av} = \frac{\Psi_{di}^{av}}{J_{wr}} * u_{wr} + \frac{\Psi_{di}^{av}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$). Where Ψ_{di}^{av} is the data size that consists of the vehicle video camera device captured data amount for autonomous vehicle application.

β_{pvp}^{av} is the primary virtual network function processing delay at the virtual network function processing node server for the autonomous vehicle application ($\beta_{pvp}^{av} = \frac{q_{nfw}^{av}+q_{atn}^{av}+q_{npdi}^{ia}}{\Lambda_{dd}} + \frac{\Psi_{tvpd}^{av}}{J_{lr}} * u_{lr}$).

q_{nfw}^{av}, q_{atn}^{av}, and q_{npdi}^{av} are the virtual network function workload amounts for firewall, NAT, and DPI for autonomous vehicle applications during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ_{tvpd}^{av} is the exchanged data amount during virtual network processing operation execution.

β_{dwp}^{av} is the digital work node (at the MEC server) based offloaded data processing (i.e., traffic light detection result,road object detection result, and vehicle action such as moving or stop action generation based on image detection) delay ($\beta_{dwp}^{av} = \frac{q_{dwp}^{av}}{\Lambda_{dd}}$). Where q_{dwp}^{av} is the number of workloads for the offloaded data processing at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β_{rdt}^{av} is the digital work node generated resultant data dispatch delay to vehicle for action ($\beta_{rdt}^{av} = \frac{\Psi_{rdt}^{av}}{J_{wr}} * u_{wr} + \frac{\Psi_{rdt}^{av}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$). Where Ψ_{rdt}^{av} is the output data size that consists of the digital work node generated resultant data for autonomous vehicle applications.

β_{svp}^{av} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the vehicles device ($\beta_{svp}^{av} = \frac{q_{npdi}^{av}+q_{nfw}^{av}+q_{nid}^{av}}{\Lambda_{dd}} + \frac{\Psi_{stvd}^{av}}{J_{lr}} * u_{lr}$).

q_{npdi}^{av}, q_{nfw}^{av}, and q_{nid}^{av} are the virtual network function workload amounts for DPI, firewall, and intrusion detection-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ_{stvd}^{av} is the secondary exchanged network function processing data amount during virtual network processing operation during autonomous vehicle application. β_{xpr}^{av} is the receiver vehicle device-based final digital worker processed result visualization delay ($\beta_{xpr}^{av} = \frac{q_{xpr}^{av}}{\Lambda_{pd}}$). q_{xpr}^{av} is the workload for the receiver XR or vehicle device for final work completion result visualization.

Next, we will present a non-metaverse-based traditional broadband-based personal data transfer application execution delay. The work completion delay for traditional broadband-based data transfer application execution β_{tb}^{k} is given by:

$$\beta_{tb}^{k} = \sum_{k=1}^{z}(\beta_{cim}^{tb} + \beta_{trd}^{tb} + \beta_{tmp}^{tb} + \beta_{sip}^{tb} + \beta_{xpu}^{tb} + \beta_{xdu}^{tb} + \beta_{pvp}^{tb} + \beta_{dwp}^{tb} + \beta_{rdt}^{tb} + \beta_{svp}^{tb} + \beta_{xpr}^{tb}) \quad (16)$$

β_{cim}^{tb} is the initial device readyness and internet connectivity delay for the user device. β_{trd}^{tb} is the traditional broadband-based data transfer application execution request dispatch delay to the nearby task manager. β_{tmp}^{tb} is the digital task manager-based traditional broadband-based data transfer application instruction generation delay for the workers. β_{sip}^{tb} is a traditional broadband-based data transfer application processing work instruction transfer delay from the digital task manager to workers. β_{cim}^{tb}, β_{trd}^{tb}, β_{tmp}^{tb}, and β_{sip}^{tb} are calculated by using the similar calculation formula as β_{cim}^{k}, β_{trd}^{k}, β_{tmp}^{k}, and β_{sip}^{k} (see first XR based gaming application discussion).

β_{xpu}^{tb} is the user XR device or mobile phone-based personal data capture (i.e., own image or message) operation delay ($\beta_{xpu}^{tb} = \frac{q_{xpu}^{tb}}{\Lambda_{pd}}$). q_{xpu}^{tb} is the workload for user XR device or mobile phone-based personal data capture (i.e., own image or message) operation. Λ_{pd} is the work processing power or CPU speed for the user device.

β_{xdu}^{tb} is the user device captured data upload delay to the digital worker device at the MEC server ($\beta_{xdu}^{tb} = \frac{\Psi_{di}^{tb}}{J_{wr}} * u_{wr} + \frac{\Psi_{di}^{tb}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$). Where Ψ_{di}^{tb} is the data size that consists of the user's XR device or mobile phone captured personal data (i.e., personal image or message).

β_{pvp}^{tb} is the primary virtual network function processing delay at the virtual network function processing node server for the traditional broadband-based data transfer application ($\beta_{pvp}^{tb} = \frac{q_{nfw}^{tb}+q_{atn}^{tb}+q_{npdi}^{tb}}{\Lambda_{dd}} + \frac{\Psi_{tvpd}^{tb}}{J_{lr}} * u_{lr}$).

q_{nfw}^{tb}, q_{atn}^{tb}, and q_{npdi}^{tb} are the virtual network function workload amounts for firewall, NAT, and DPI for traditional broadband-based data transfer applications during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ_{tvpd}^{tb} is the exchanged data amount during virtual network processing operation execution.

β_{dwp}^{tb} is the digital work node (at the MEC server) based offloaded data processing (i.e., compression, encoding, receiver location selection) delay ($\beta_{dwp}^{tb} = \frac{q_{dwp}^{tb}}{\Lambda_{dd}}$). Where β_{dwp}^{tb} is the number of workloads for the offloaded data processing at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β_{rdt}^{tb} is the digital work node generated resultant data dispatch delay to receiver user device ($\beta_{rdt}^{tb} = \frac{\Psi_{rdt}^{tb}}{J_{wr}} * u_{wr} + \frac{\Psi_{rdt}^{tb}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$). Where Ψ_{rdt}^{tb} is the output data size that consists of the digital work node-generated resultant data for traditional broadband-based data transfer applications.

β_{svp}^{tb} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the receiver device ($\beta_{svp}^{tb} = \frac{q_{npdi}^{tb}+q_{nfw}^{tb}+q_{atn}^{tb}}{\Lambda_{dd}} + \frac{\Psi_{stvd}^{tb}}{J_{lr}} * u_{lr}$).

q_{npdi}^{tb}, q_{nfw}^{tb}, and q_{atn}^{tb} are the virtual network function workload amounts for DPI, firewall, and NAT-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ_{stvd}^{tb} is the secondary exchanged network function processing data amount during virtual network processing operation during traditional broadband-based data transfer applications.

Next, we will present a non-metaverse-based caching application execution delay. The work completion delay for caching application execution β_{vc}^{k} is given by:

$$\beta_{vc}^{k} = \sum_{k=1}^{z}(\beta_{cim}^{vc} + \beta_{trd}^{vc} + \beta_{tmp}^{vc} + \beta_{sip}^{vc} + \beta_{xpu}^{vc} + \beta_{xdu}^{vc} + \beta_{pvp}^{vc} + \beta_{dwp}^{vc} + \beta_{rdt}^{vc} + \beta_{svp}^{vc} + \beta_{xpr}^{vc}) \quad (17)$$

β_{cim}^{vc} is the initial device readyness and internet connectivity delay for the user device. β_{trd}^{vc} is the caching application execution request dispatch delay to the nearby task manager. β_{tmp}^{vc} is the digital task manager-based traditional caching application execution application instruction generation delay for the workers (MEC, user device). β_{sip}^{vc} is traditional caching application execution processing work instruction transfer delay from the digital task manager to workers. β_{cim}^{vc}, β_{trd}^{vc}, β_{tmp}^{vc}, and β_{sip}^{vc} are calculated by using the similar calculation formula as β_{cim}^{k}, β_{trd}^{k}, β_{tmp}^{k}, and β_{sip}^{k} (see first XR based gaming application discussion).

β_{xpu}^{vc} is the user XR device or mobile phone-based video file name information selection or file name capture operation delay ($\beta_{xpu}^{vc} = \frac{q_{xpu}^{vc}}{\Lambda_{pd}}$). q_{xpu}^{vc} is the workload for the user's XR device or mobile phone-based caching request data capture operation. Λ_{pd} is the work processing power or CPU speed for the user device.

β_{xdu}^{vc} is the user device captured data upload delay to the digital worker device at the MEC server ($\beta_{xdu}^{vc} = \frac{\Psi_{di}^{vc}}{J_{wr}} * u_{wr} + \frac{\Psi_{di}^{vc}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$). Where Ψ_{di}^{vc} is the data size that consists of the user XR device or mobile phone captured users caching request data.

β_{pvp}^{vc} is the primary virtual network function processing delay at the virtual network function processing node server for the video caching application ($\beta_{pvp}^{vc} = \frac{q_{nfw}^{vc}+q_{atn}^{vc}+q_{npdi}^{vc}}{\Lambda_{dd}} + \frac{\Psi_{tvpd}^{vc}}{J_{lr}} * u_{lr}$).

q_{nfw}^{vc}, q_{atn}^{vc}, and q_{npdi}^{vc} are the virtual network function workload amounts for firewall, NAT, and DPI operations during offloading data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ_{tvpd}^{vc} is the exchanged data amount during virtual network processing operation execution.

β_{dwp}^{vc} is the digital work node (at the MEC server) based offloaded data processing (i.e., cache lookup, file search, preparation for access) delay ($\beta_{dwp}^{vc} = \frac{q_{dwp}^{vc}}{\Lambda_{dd}} + H*(\beta_{ew}^{vc} + \beta_{rw}^{vc})$. Where q_{dwp}^{vc} is the number of workloads for the offloaded data processing (i.e., cache lookup, file search, preparation for access) at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server). H is the cache hit ratio (H = 0 or 1). β_{ew}^{vc} is the associated communication delay if the video file is accessed from the edge cloud server. β_{rw}^{vc} is the associated communication delay if the video file is accessed from the remote or central cloud server (similar to [133]).

β_{rdt}^{vc} is the digital work node generated resultant caching data dispatch delay to receiver user device ($\beta_{rdt}^{vc} = \frac{\Psi_{rdt}^{vc}}{J_{wr}} * u_{wr} + \frac{\Psi_{rdt}^{vc}}{J_{or}} * u_{or} + \beta_{pp}^{k} + \beta_{rwd}^{k}$). Where Ψ_{rdt}^{vc} is the output data size that consists of the digital work node-generated caching file data.

β_{svp}^{vc} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the receiver device ($\beta_{svp}^{vc} = \frac{q_{npdi}^{vc}+q_{nfw}^{vc}+q_{nid}^{tb}}{\Lambda_{dd}} + \frac{\Psi_{stvd}^{vc}}{J_{lr}} * u_{lr}$).

q_{npdi}^{vc}, q_{nfw}^{vc}, and q_{nid}^{vc} are the virtual network function workload amounts for DPI, firewall, and IDS-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ_{stvd}^{vc} is the secondary exchanged network function processing data amount during virtual network processing operation during caching-based application.

β_{xpr}^{vc} is the receiver device-based caching data visualization delay ($\beta_{xpr}^{vc} = \frac{q_{xpr}^{vc}}{\Lambda_{pd}}$). q_{xpr}^{vc} is the workload for the receiver XR or mobile phone device for final caching data visualization.

Now, this article will present a non-metaverse-based human brain-computer-wheelchair interaction application execution delay. The work completion delay for human brain-computer-wheelchair interaction-based application execution

β^{k}_{bcx} is given by:

$$\beta^{k}_{bcx} = \sum_{k=1}^{z}(\beta^{bcx}_{cim} + \beta^{bcx}_{trd} + \beta^{bcx}_{tmp} + \beta^{bcx}_{sip} + \beta^{bcx}_{xpu} + \beta^{bcx}_{xdu} + \beta^{bcx}_{pvp} + \beta^{bcx}_{dwp} + \beta^{bcx}_{rdt} + \beta^{bcx}_{svp} + \beta^{bcx}_{xpr}) \quad (18)$$

β^{bcx}_{cim} is the initial device readyness and internet connectivity delay for the user device. β^{bcx}_{trd} is the human brain-computer-wheelchair interaction application that requests dispatch delay to the nearby task manager. β^{bcx}_{tmp} is the digital task manager-based human brain-computer-wheelchair interaction application instruction generation delay for the workers (MEC, brain sensors, wheelchair, user device). β^{bcx}_{sip} is human brain-computer-wheelchair interaction application processing work instruction transfer delay from the digital task manager to workers. β^{bcx}_{cim}, β^{bcx}_{trd}, β^{bcx}_{tmp}, and β^{bcx}_{sip} are calculated by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{bcx}_{xpu} is the user brain sensor-based data capture (i.e., brain signal) operation delay ($\beta^{bcx}_{xpu} = \frac{q^{bcx}_{xpu}}{\Lambda_{pd}}$). q^{bcx}_{xpu} is the workload for user brain sensor-based data capture operations. Λ_{pd} is the work processing power or CPU speed for the user brain sensor device.

β^{bcx}_{xdu} is the user brain sensor device captured data upload delay to digital worker device at the MEC server ($\beta^{bcx}_{xdu} = \frac{\Psi^{bcx}_{di}}{J_{wr}} * u_{wr} + \frac{\Psi^{bcx}_{di}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{bcx}_{di} is the data size that consists of the user brain sensor device captured data.

β^{bcx}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the human brain. computer-wheelchair interaction application ($\beta^{bcx}_{pvp} = \frac{q^{bcx}_{nfw}+q^{bcx}_{atn}+q^{bcx}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{bcx}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{bcx}_{nfw}, q^{bcx}_{atn}, and q^{bcx}_{npdi} are the virtual network function workload amounts for firewall, NAT, and DPI for human brain-computer-wheelchair interaction application execution during offloaded data on the MEC server, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{bcx}_{tvpd} is the exchanged data amount during virtual network processing operation execution.

β^{bcx}_{dwp} is the digital work node (at the MEC server) based offloaded data processing (i.e., brain signal to instruction or action generation) delay ($\beta^{bcx}_{dwp} = \frac{q^{bcx}_{dwp}}{\Lambda_{dd}}$). Where q^{bcx}_{dwp} is the number of workloads for the offloaded data processing at the digital worker node. Λ_{dd} is the workload processing power at the digital work processing node (MEC server).

β^{bcx}_{rdt} is the digital work node generated resultant data dispatch delay to receiver wheelchair device ($\beta^{bcx}_{rdt} = \frac{\Psi^{bcx}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{bcx}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd}$). Where Ψ^{bcx}_{rdt} is the output data size that consists of the digital work node generated resultant data for the wheelchair operation.

β^{bcx}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the receiver wheelchair device ($\beta^{bcx}_{svp} = \frac{q^{bcx}_{npdi}+q^{bcx}_{nfw}+q^{bcx}_{atn}}{\Lambda_{dd}} + \frac{\Psi^{bcx}_{stvd}}{J_{lr}} * u_{lr}$).

q^{bcx}_{npdi}, q^{bcx}_{nfw}, and q^{bcx}_{atn} are the virtual network function workload amounts for DPI, firewall, and NAT-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{bcx}_{stvd} is the secondary exchanged network function processing data amount during virtual network processing operation during human brain-computer-wheelchair interaction application execution.

β^{bcx}_{xpr} is the receiver wheelchair device-based final instrauction execution (e.g., move forward) delay ($\beta^{bcx}_{xpr} = \frac{q^{bcx}_{xpr}}{\Lambda_{pd}}$). q^{bcx}_{xpr} is the workload for the receiver wheelchair device for final work execution.

Next, this paper will discuss a non-metaverse-based charging station for vehicle electric energy transfer applications. The work completion delay for charging stations to vehicles electric energy transfer application execution β^{k}_{vet} is given by:

$$\beta^{k}_{vet} = \sum_{k=1}^{z}(\beta^{vet}_{cim} + \beta^{vet}_{trd} + \beta^{vet}_{tmp} + \beta^{vet}_{sip} + \beta^{vet}_{pvp} + \beta^{vet}_{xmu} + \beta^{vet}_{csp} + \beta^{vet}_{rdt} + \beta^{vet}_{svp} + \beta^{vet}_{xpr}) \quad (19)$$

β^{vet}_{cim} is the initial device readyness and internet connectivity delay for user devices and vehicles. β^{vet}_{trd} is the charging station to vehicles electric energy transfer application request dispatch delay to nearby task manager. β^{vet}_{tmp} is the digital task manager-based charging station to vehicles electric energy transfer application instruction generation (i.e., charging time selection for vehicles and charging station selection) delay for the workers (charging station, vehicles, user device). β^{vet}_{sip} is an energy transfer application processing work instruction transfer delay from the digital task manager to workers (vehicle, charging station). β^{vet}_{cim}, β^{vet}_{trd}, β^{vet}_{tmp}, and β^{vet}_{sip} are measured by using the similar calculation formula as β^{k}_{cim}, β^{k}_{trd}, β^{k}_{tmp}, and β^{k}_{sip} (see first XR based gaming application discussion).

β^{vet}_{pvp} is the primary virtual network function processing delay at the virtual network function processing node server for the charging station to vehicles electric energy transfer application ($\beta^{vet}_{pvp} = \frac{q^{vet}_{nfw}+q^{vet}_{atn}+q^{vet}_{npdi}}{\Lambda_{dd}} + \frac{\Psi^{vet}_{tvpd}}{J_{lr}} * u_{lr}$).

q^{vet}_{nfw}, q^{vet}_{atn}, and q^{vet}_{npdi} are the virtual network function workload amounts for firewall, NAT, and DPI for charging station to vehicle electric energy transfer application execution during MEC instruction reception data in the user vehicle device, respectively. Λ_{dd} is the virtual network function processing device at the digital worker node or virtual network function processing node. Ψ^{vet}_{tvpd} is the exchanged data amount during virtual network function processing operation execution.

β^{vet}_{xmu} is the vehicle device-based movement operation execution (e.g., move to the charging station) delay ($\beta^{vet}_{xmu} = \frac{d^{vet}_{xmu}}{\Lambda_{pm}}$). d^{vet}_{xmu} is the distance between the vehicle and charging station. Λ_{pm} is the movement speed for the vehicles.

β^{vet}_{csp} is the receiver vehicle-based charging operation execution (e.g., at the charging station) delay ($\beta^{vet}_{csp} = \frac{x_{rr}-x_{aa}+x_{bn}*x_{th}}{a_{cr}}$). Where $x_{rr}, x_{aa}, x_{bn}, x_{th}$, and a_{cr} are required charges for vehicles, already present charge availability at the vehicle, threshold value for vehicle battery depletion, capacity of vehicle battery, and rate of charging for vehicles (per second), respectively.

β^{vet}_{rdt} is the work completion message data dispatch delay to receiver vehicles device from charging station ($\beta^{vet}_{rdt} =$

$\frac{\Psi^{vet}_{rdt}}{J_{wr}} * u_{wr} + \frac{\Psi^{vet}_{rdt}}{J_{or}} * u_{or} + \beta^{k}_{pp} + \beta^{k}_{rwd})$. Where Ψ^{vet}_{rdt} is the output data size (e.g., work completion message).

β^{vet}_{svp} is the secondary virtual network function processing delay at the virtual network processing work node before the resultant data reception at the receiver vehicle's device ($\beta^{vet}_{svp} = \frac{q^{vet}_{npdi}+q^{vet}_{nfw}+q^{vet}_{nid}}{\Lambda_{dd}} + \frac{\Psi^{vet}_{stvd}}{J_{lr}} * u_{lr}$).

q^{vet}_{npdi}, q^{vet}_{nfw}, and q^{vet}_{nid} are the virtual network function workload amounts for DPI, firewall, and IDS-based VNF operations, respectively. Λ_{dd} is the virtual network function processing device or worker node processing speed (CPU speed). Ψ^{vet}_{stvd} is the secondary exchanged network function processing data amount during virtual network function processing operation during harging station to vehicle electric energy transfer application execution. β^{vet}_{xpr} is the receiver device-based final result display delay ($\beta^{vet}_{xpr} = \frac{q^{vet}_{xpr}}{\Lambda_{pd}}$). q^{vet}_{xpr} is the workload for the receiver device for final work completion result display.

Next, this article will present multiverse (multiple metaverse platform-based) avatar socialization and NFT trading application execution delays. The work completion delay for avatar socialization and NFT trading application execution β^{k}_{snt} is given by:

$$\beta^{k}_{snt} = \sum_{k=1}^{z} \beta^{k}_{sa} + \beta^{k}_{bnas}$$

Where β^{k}_{sa} is the avatar socialization application work completion delay. β^{k}_{bnas} is the blockchain and NFT (non-fungible token) transfer-based visual art selling application work completion delay.

β^{k}_{sa} calculation is described in equation 9 (see previously discussed section on avatar socialization application work completion delay calculation). β^{k}_{bnas} calculation is described in equation 12 (see previously discussed section on blockchain and NFT (non-fungible token) transfer-based visual art selling application work completion delay calculation).

Finally, this article will discuss another multiverse (multiple metaverse platform-based)-based virtual clothing and a virtual tour of museum application execution delays. The work completion delay β^{k}_{cmt} for virtual clothing (via avatar) and virtual tour of museum applications (via avatar) is calculated by:

$$\beta^{k}_{cmt} = \sum_{k=1}^{z} \beta^{k}_{vce} + \beta^{k}_{mt}$$

Where β^{k}_{vce} is the virtual clothing experience application work completion delay. β^{k}_{mt} is the virtual museum tour application work completion delay.

β^{k}_{vce} calculation is described in equation 7 (see previously discussed section of XR-based virtual clothing experience work completion delay calculation). β^{k}_{mt} calculation is briefed in equation 6 (see previously discussed section of XR-based virtual museum tour experience application work completion delay calculation).

B. Economic cost for users

The economic cost for users (γ^{k}_{uec}) concerning different 6G application (metaverse, multiverse, and non-metaverse application) execution can be calculated by taking the summation of users financial cost for bandwidth resource, local user device or physical device resource use, virtual network function processing server resource use, resource waiting activity, cloud/caching/blockchain server resource usage. γ^{k}_{uec} is analyzed by:

$$\begin{aligned}\gamma^{k}_{uec} = &\sum_{k=1}^{\xi_{hc}}(\varsigma^{k}_{bw} * \beta^{k}_{et} + \varsigma^{k}_{bw} * \beta^{k}_{er}) + \sum_{i=1}^{\xi_{hc}}(\varsigma^{k}_{uo} * \beta^{k}_{vc}\\ &+ \varsigma^{k}_{sh} * \beta^{k}_{vc} + \varsigma^{k}_{sp} * \beta^{k}_{vc}) + \sum_{k=1}^{\xi_{hc}}(\varsigma^{i}_{ew} * \beta^{k}_{ew} + \varsigma^{k}_{ud} * \beta^{k}_{pd}) +\\ &\sum_{k=1}^{\xi_{mc}}(\varsigma^{k}_{bw} * \beta^{k}_{et} + \varsigma^{k}_{bw} * \beta^{k}_{er}) + \sum_{i=1}^{\xi_{mc}}(\varsigma^{k}_{uo} * \beta^{k}_{vc}\\ &+ \varsigma^{k}_{sh} * \beta^{k}_{vc} + \varsigma^{k}_{sp} * \beta^{k}_{vc}) + \sum_{k=1}^{\xi_{mc}}(\varsigma^{i}_{ew} * \beta^{k}_{ew} + \varsigma^{k}_{ud} * \beta^{k}_{pd}) +\\ &\sum_{k=1}^{\xi_{lc}}(\varsigma^{k}_{bw} * \beta^{k}_{et} + \varsigma^{k}_{bw} * \beta^{k}_{er}) + \sum_{i=1}^{\xi_{lc}}(\varsigma^{k}_{uo} * \beta^{k}_{vc}\\ &+ \varsigma^{k}_{sh} * \beta^{k}_{vc} + \varsigma^{k}_{sp} * \beta^{k}_{vc}) + \sum_{k=1}^{\xi_{lc}}(\varsigma^{i}_{ew} * \beta^{k}_{ew} + \varsigma^{k}_{ud} * \beta^{k}_{pd})\end{aligned} \tag{20}$$

ξ_{hc}, ξ_{mc}, and ξ_{lc} are the total number of high-time critical requirement-based, medium-time critical requirement-based, and low-time critical requirement-based 6G applications (e.g., multiverse-based, metaverse-based, and non-metaverse-based applications), respectively. The total number of applications is z = ξ_{hc}+ξ_{mc}+ξ_{lc}.

ς^{k}_{bw}, ς^{k}_{ud}, and ς^{k}_{ew} are the economic expenditure average cost (per milisecond) for users concerning bandwidth usage (for data transfer and receive), physical local device usage for local device-based work processing (e.g., mobile phone, robot, sensor, XR device), and resource access waiting activity, respectively. ς^{k}_{uo}, ς^{k}_{sh}, and ς^{k}_{sp} are the economic expenditure average cost (per milisecond) for users concerning user-owned virtual network function processing and digital work node resource usage (e.g., MEC server, VNF server, blockchain server), sharing neighbor-owned virtual network function processing and digital work node resource usage, and service provider-owned virtual network function processing and digital work node resource usage, respectively.

β^{k}_{et}, β^{k}_{er}, β^{k}_{pd}, β^{k}_{vc}, and β^{k}_{ew} are the application work completion delays concerning application data transmission activity, application data receive activity, physical device (e.g., user device, XR device, robot, video sensor)-based application work processing activity, digital device-based application work processing (e.g., cloud server-based work processing, virtual network function processing) activity, and resource access waiting activity, respectively.

C. Sucessful task completion ratio

The successful task completion ratio (ϵ^{k}_{stcr}) can be ascertained by taking the ratio of the total application number that can fulfill the deadline requirements during execution and the total arrived application number (e.g., metaverse, multiverse, and non-metaverse). The successful task completion ratio ϵ^{k}_{stcr} is measured by $\epsilon^{k}_{stcr} = \frac{\sum_{k=1}^{z}\xi^{k}_{ta} - \sum_{k=1}^{z}\xi^{k}_{ns}}{\sum_{k=1}^{z}\xi^{k}_{ta}} = \frac{\sum_{k=1}^{z}\xi^{k}_{ts}}{\sum_{k=1}^{z}\xi^{k}_{ta}}$.

Where ξ_{ta}^k, ξ_{ns}^k, and ξ_{ts}^k are the total arrived application number, total unsucessful application that cannot satisfy the deadline limit, and total successful application that can satisfy the deadline limit, respectively.

D. Energy expenditure of users devices

The energy expenditure of user devices (G_{ee}^k) for the total k number of 6G applications (metaverse, multiverse, and non-metaverse application execution) can be measured by taking the summation of users different energy expenditure values concerning application data transfer activity, application data receive activity, physical and digital work node-based application workload processing, and waiting activity. G_{ee}^k can be analyzed as follows:

$$\begin{aligned}G_{ee}^k = \sum_{k=1}^{\xi_{mv}}(y_{et}^k*\beta_{et}^k + y_{er}^k*\beta_{er}^k + y_{pd}^k*\beta_{pd}^k)\\ + \sum_{k=1}^{\xi_{mv}}(y_{vc}^k*\beta_{vc}^k + y_{ew}^k*\beta_{ew}^k)+\\ \sum_{k=1}^{\xi_{me}}(y_{et}^k*\beta_{et}^k+y_{er}^k*\beta_{er}^k+y_{pd}^k*\beta_{pd}^k)+\sum_{k=1}^{\xi_{me}}(y_{vc}^k*\beta_{vc}^k+y_{ew}^k*\beta_{ew}^k)+\\ \sum_{k=1}^{\xi_{nme}}(y_{et}^k*\beta_{et}^k+y_{er}^k*\beta_{er}^k+y_{pd}^k*\beta_{pd}^k)+\sum_{k=1}^{\xi_{nme}}(y_{vc}^k*\beta_{vc}^k+y_{ew}^k*\beta_{ew}^k)\end{aligned} \tag{21}$$

ξ_{mv}, ξ_{me}, and ξ_{nme} are the total number of executed multiverse-based 6G applications, metaverse-based 6G applications, and non-metaverse-based 6G applications, respectively. The total number of applications is z = ξ_{mv}+ξ_{me}+ξ_{nme}.

y_{et}^k, y_{er}^k, y_{pd}^k, y_{vc}^k, and y_{ew}^k are the energy expenditure average cost (per milisecond) for user host devices concerning application data transmission activity, application data receive activity, physical device (e.g., user device, XR device, robot, video sensor)-based application work processing activity, digital device-based application work processing (e.g., cloud server-based processing, virtual network function processing) activity, and resource access waiting activity, respectively.

β_{et}^k, β_{er}^k, β_{pd}^k, β_{vc}^k, and β_{ew}^k are the application work completion delays concerning application data transmission activity, application data receive activity, physical device (e.g., user device, XR device, robot, video sensor)-based application work processing activity, digital device-based application work processing (e.g., cloud server-based work processing, virtual network function processing) activity, and resource access waiting activity, respectively.

E. Total throughput value for the network

The total network throughput value χ_{nwt}^k is ascertained by taking the ratio of exchanged data size total amount during application execution and application execution makespan time delay (i.e., maximum time delay for all arrived application execution).

χ_{nwt}^k is described by: $\chi_{nwt}^k = \frac{\sum_{k=1}^{z}(\Psi_{dit}^k+\Psi_{dot}^k+\Psi_{dod}^k)}{\sum_{k=1}^{z}\beta_{ts}^k}$. Where Ψ_{dit}^k is the arrived application input data size total value. Ψ_{dot}^k is the arrived application output data size total value. Ψ_{dod}^k is the arrived applications other exchanged data size total value during different operation execution.

β_{ts}^k is the application's maximum timespan delay. β_{ts}^k is computed by:
β_{ts}^k=max$\{\beta_{asd}^1, \beta_{asd}^2, ..\beta_{asd}^k\}$, $k\in 1,2,..z$. z is the total application count number. β_{asd}^k is the end-to-end application service delivery delay (please see Section 5.1 for details).

F. Total profit for the different service providers

The total profit value concerning different service providers ($\gamma_{tpv}^k = \sum_{k=1}^{z}\gamma_{uec}^k - \gamma_{spec}^k$) can be measured by taking the difference between collected economic expenditure or revenue from users (γ_{uec}^k) and service provider cost (γ_{spec}^k) for different services (i.e., resource buy, service delivery, maintenance). γ_{spec}^k is the service provider's cost for different 6G application service delivery. γ_{spec}^k can be analyzed as follows:

$$\begin{aligned}\gamma_{spec}^k = \sum_{k=1}^{\xi_{hc}}(\varsigma_{sbw}^k*\beta_{et}^k + \varsigma_{sbw}^k*\beta_{er}^k) + \sum_{i=1}^{\xi_{hc}}(\varsigma_{suo}^k*\beta_{vc}^k\\ + \varsigma_{ssh}^k*\beta_{vc}^k + \varsigma_{ssp}^k*\beta_{vc}^k) + \sum_{k=1}^{\xi_{hc}}(\varsigma_{sew}^i*\beta_{ew}^k + \varsigma_{sud}^k*\beta_{pd}^k)+\\ \sum_{k=1}^{\xi_{mc}}(\varsigma_{sbw}^k*\beta_{et}^k + \varsigma_{sbw}^k*\beta_{er}^k) + \sum_{i=1}^{\xi_{mc}}(\varsigma_{suo}^k*\beta_{vc}^k\\ + \varsigma_{ssh}^k*\beta_{vc}^k + \varsigma_{ssp}^k*\beta_{vc}^k) + \sum_{k=1}^{\xi_{mc}}(\varsigma_{sew}^i*\beta_{ew}^k + \varsigma_{sud}^k*\beta_{pd}^k)+\\ \sum_{k=1}^{\xi_{lc}}(\varsigma_{sbw}^k*\beta_{et}^k + \varsigma_{sbw}^k*\beta_{er}^k) + \sum_{i=1}^{\xi_{lc}}(\varsigma_{suo}^k*\beta_{vc}^k\\ + \varsigma_{ssh}^k*\beta_{vc}^k + \varsigma_{ssp}^k*\beta_{vc}^k) + \sum_{k=1}^{\xi_{lc}}(\varsigma_{sew}^i*\beta_{ew}^k + \varsigma_{sud}^k*\beta_{pd}^k)\end{aligned} \tag{22}$$

ς_{sbw}^k, ς_{sud}^k, and ς_{sew}^k are the economic expenditure average cost (per milisecond) for different service providers concerning bandwidth service (i.e., resource buying cost, maintainance, and electricity cost) delivery to users, physical local device usage service delivery for application execution, and resource access waiting operation, respectively. The total number of applications is z = ξ_{hc}+ξ_{mc}+ξ_{lc}. ς_{suo}^k, ς_{ssh}^k, and ς_{ssp}^k are the economic expenditure average cost (per milisecond) for service providers concerning user-owned virtual network function processing/digital work node resource usage service delivery (e.g., MEC server, VNF server, blockchain server), sharing neighbor-owned virtual network function processing/digital work node resource usage, and service provider-owned virtual network function processing/digital work node resource service delivery (i.e., resource buying cost, maintenance cost, and electricity cost), respectively.

β_{et}^k, β_{er}^k, β_{pd}^k, β_{vc}^k, and β_{ew}^k are the users application work completion delays concerning application data transmission activity, application data receive activity, physical device (e.g., user device, XR device, robot, video sensor) based application work processing activity, digital device based application work processing (e.g., cloud server based work processing, virtual network function processing) activity, and resource access waiting activity, respectively.

G. User total benefit or welfare ratio

The total user benefit ratio or welfare ratio (v_{wr}^k) is measured by taking the summation of time benefit ratio (v_{bt}^k), energy benefit ratio (v_{bee}^k), and economic cost ratio (v_{bec}^k). v_{wr}^k is given by:

$$v_{wr}^k = \sum_{k=1}^{z} v_{bt}^k + v_{bee}^k + v_{bec}^k = \frac{\sum_{k=1}^{z} \beta_{td}^k - \beta_{ts}^k}{\sum_{k=1}^{z} \beta_{td}^k} + \frac{\sum_{k=1}^{z} G_{ed}^k - G_{ee}^k}{\sum_{k=1}^{z} G_{ed}^k} + \frac{\sum_{k=1}^{z} \gamma_{ecb}^k - \gamma_{uec}^k}{\sum_{k=1}^{z} \gamma_{ecb}^k} \tag{23}$$

Where β_{td}^k, G_{ed}^k, and γ_{ecb}^k are the host user application execution deadline value, energy expenditure budget, and economic cost budget for multiple application executions, respectively. β_{ts}^k, G_{ee}^k, and γ_{uec}^k are the application execution work completion makespan delay, energy expenditure for users, and economic cost for users for multiple application executions, respectively.

H. Average unused energy and network lifetime

The unused user device energy, or leftover energy, G_{ve}^k can be calculated by investigating the ratio of leftover energy for the users (the difference between total user energy at the beginning of the simulation and energy cost for users per simulation round) and total users within a network.

G_{ve}^k is ascertained by:
$G_{ve}^k = \frac{\sum_{k=1}^{z} G_{tt}^k - k * G_{auec}^k * \eta}{\sum_{k=1}^{z} \Phi_{un}^k}$.
Where G_{tt}^k is the total energy summation value that includes all users device energy. k is the executed application number per simulation instance. G_{auec}^k is the user's average energy expenditure per simulation round. η is the total simulation instance or round. Φ_{un}^k is the total user device number within a network.

The network lifetime (δ_{nl}^k) can be measured by determining the ratio of the the total sum of initial energy that includes all user deviceenergy at the beginning of the simulation (G_{tt}^k) and the energy expenditure of the user device for application execution per simulation round ($G_{tue}^k = k * G_{auec}^k * \eta$).

I. User survivability ratio

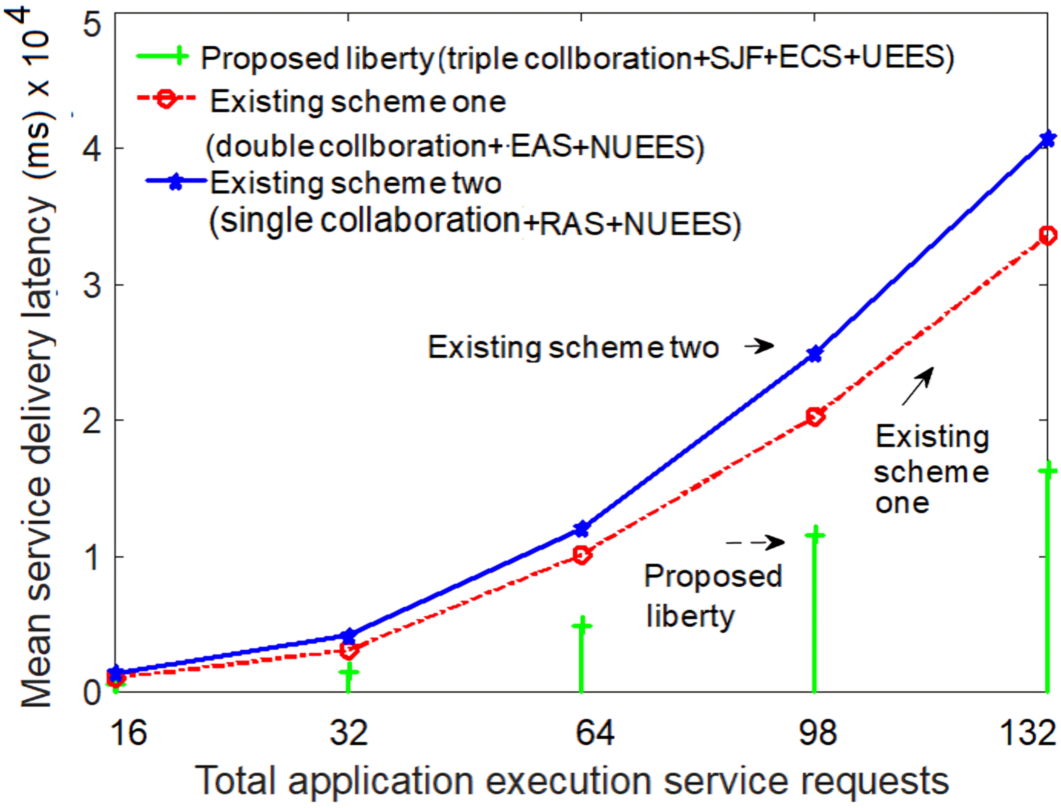
User survivability ratio (Θ_{usr}^k) is ascertained by measuring the ratio of alive or non-dead user device number after the completion of the simulation round and total user device within a network (at the beginning of the the simulation). Θ_{usr}^k is measured by:
$\Theta_{usr}^k = \sum_{\eta=1}^{\nu} \frac{\Phi_{un}^k - \Phi_{dn}^k}{\Phi_{un}^k} = \Phi_{un}^k - \sum_{\eta=1}^{\nu} \frac{G_{tue}^k}{G_{uie}}$.
Where Φ_{un}^k is the total user device number. Φ_{dn}^k is the non-alive device number after the completion of a specific simulation round. η is the simulation round number. G_{uie} is the initial energy of a user device. G_{tue}^k is the energy expenditure of the user device per simulation round.

V. Experimental Results and Analysis

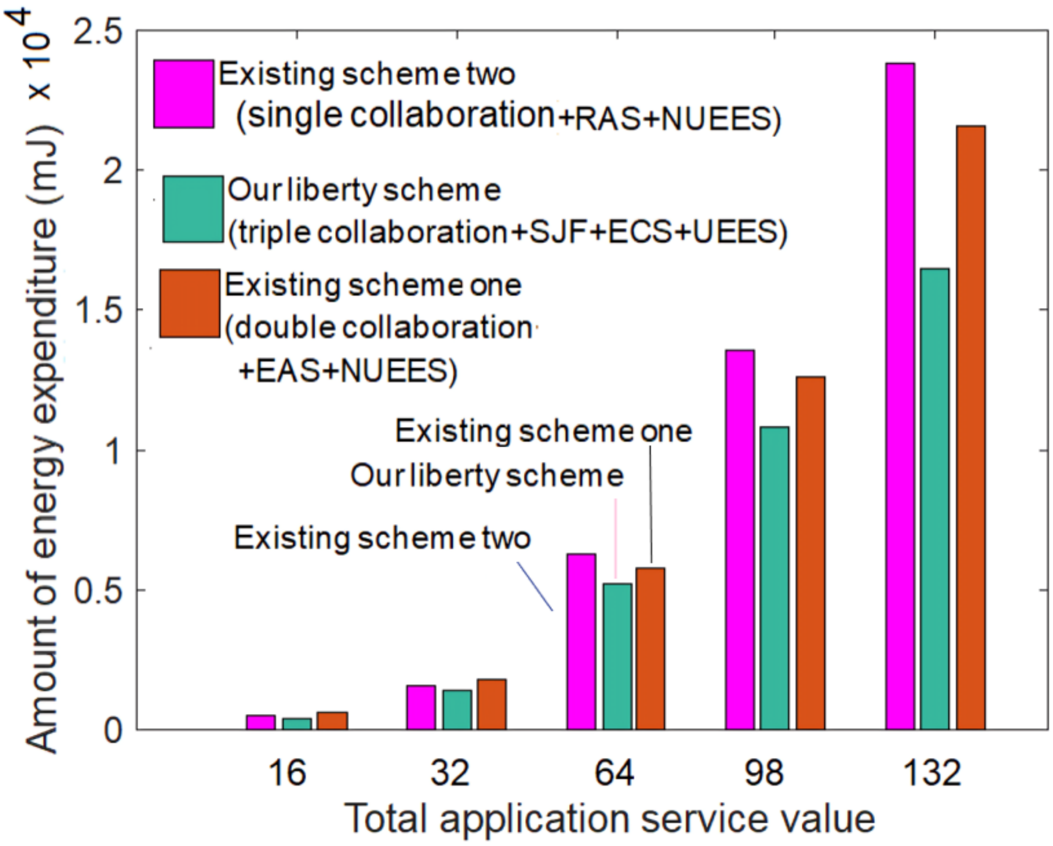
This section presents the experimental results and analysis of the proposed liberty scheme, which includes resource management, application, and worker scheduling strategies for multiverse, metaverse, and non-metaverse-based application execution over 6G edge computing networks.

The proposed liberty-based resource management and worker selection scheme (triple collaboration+SJF+ECS+UEES) has the following key features: minimum service delivery delay, economic cost saving (ECS), and user energy expenditure cost saving (UEES) based network resource and worker selection (physical worker, digital worker, virtual network function processing server, and communication resources) for application execution, the shortest deadline application first-based multi The proposed liberty scheme is compared to two existing schemes to determine which is superior. The existing scheme one (double collaboration+EAS+NUEES) selects the best resources with maximum power by verifying both user and service provider owned digital resources along with earliest application arrival first served based scheduling (EAS) and non-user energy saving based application scheduling (e.g., [70], [38]). The existing scheme two (single collaboration+EAS+NUEES) selects the best resources casually by verifying only service provider-owned resources along with random application and worker scheduling (RAS) and non-user energy-saving (NUEES)-based scheduling (e.g., [37], [137]). This article employs a simulation framework built on MATLAB. The total number of applications (multiverse, metaverse, and non-multiverse-based) ranges from 16 to 132, and is determined at random. The application execution deadline has a time limit ranging from 1900 to 57,000 milliseconds. This paper generates multiverse, metaverse, and non-metaverse application requests with various data sizes and workloads at random. However, application resource allocation (digital worker, physical worker, and VNF server) prioritizes the shortest deadline. The digital worker device and the user physical device have maximum computation work processing speeds of 4.5 GHz and 2000 MHz, respectively. Tables I-V contain experimental values for simulation, such as input and output data size for various applications, workload associated with physical workers, digital workers, and VNF processing servers, wired and wireless link data rates, energy expenditure costs, economic costs, and service provider costs. This work considers the following metaverse applications: XR-based immersive gaming, avatar-based virtual tours of museums, avatar-based virtual clothing experiences, XR-based remote healthcare, socialization via avatar interaction, hologram-based remote presence for video conferencing, federated learning-based surveillance applications, NFT (non-fungible token) and blockchain-enabled visual art selling, and digital twin-based robot selection for manufacturing jobs. Non-metaverse applications include human-digital worker-cobot-video sensor industrial automation jobs, autonomous driving applications, traditional broadband-based data exchange applications, video file caching applications, brain sensor-digital worker-wheelchair interaction-based biological application porcessing, and charging station-to-electric vehicle energy transfer applications. The multiverse application combines avatar socialization and NFT-based trading, as well as a virtual museum tour and virtual clothing experience delivered via XR and haptic devices.

Figure 3(a) depicts the relationship between the mean

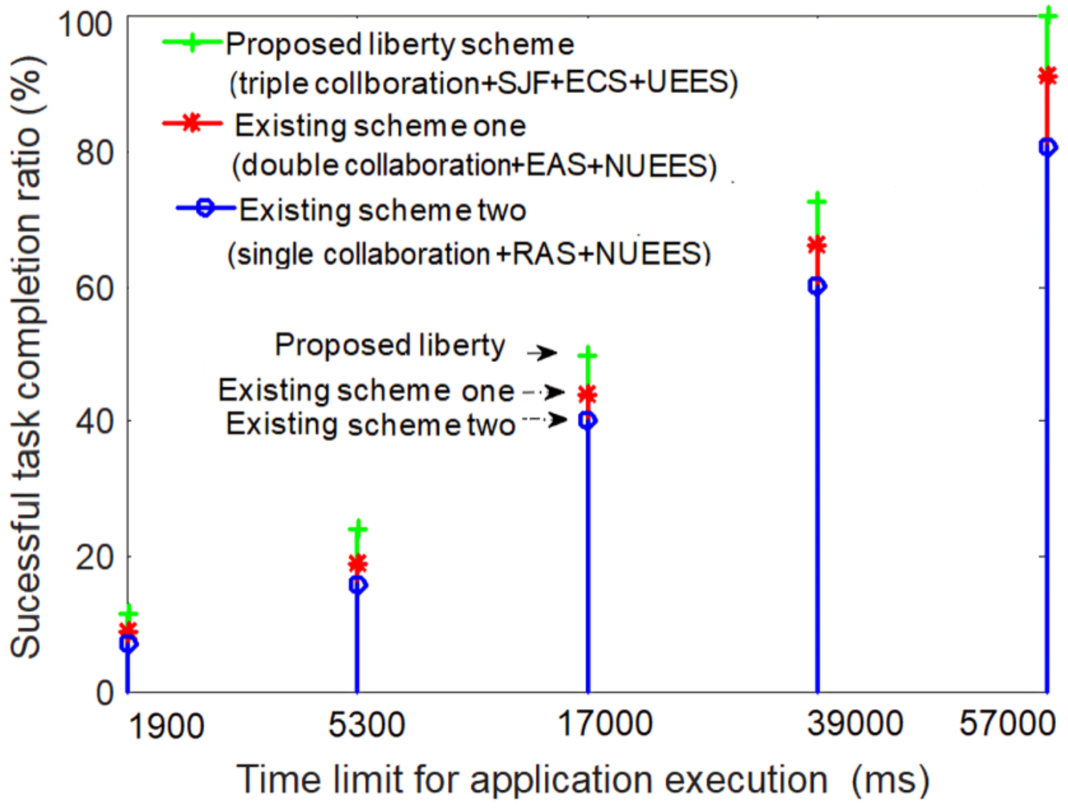


(a)

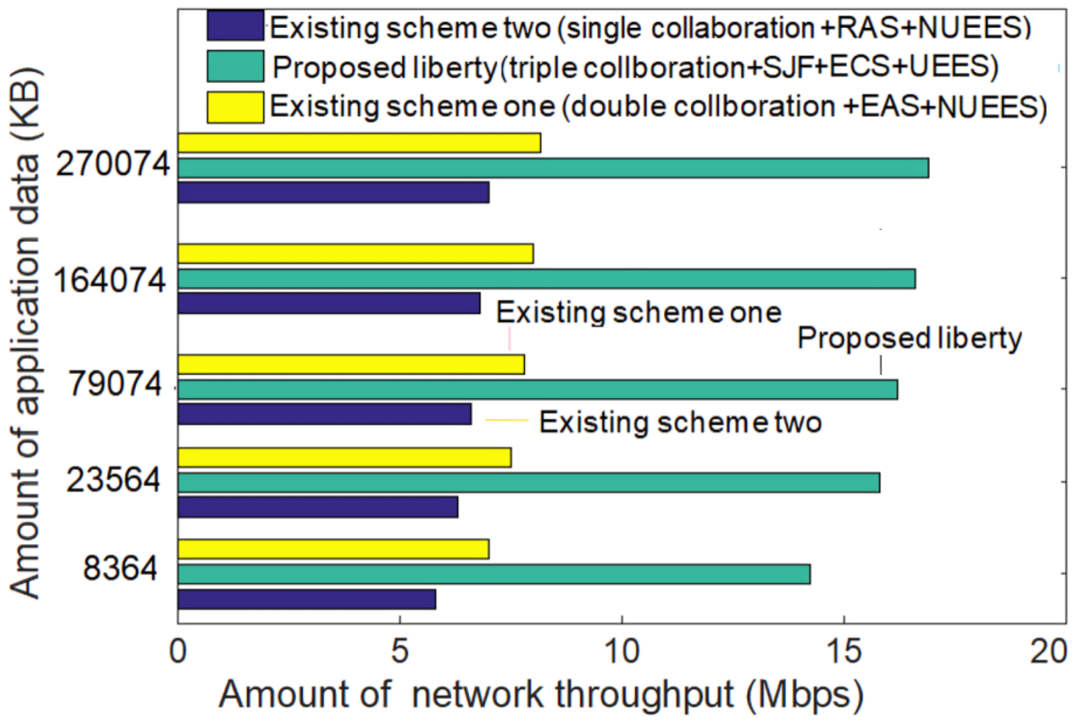


(b)

Fig. 3. Mean application service delivery latency and energy expenditure of users

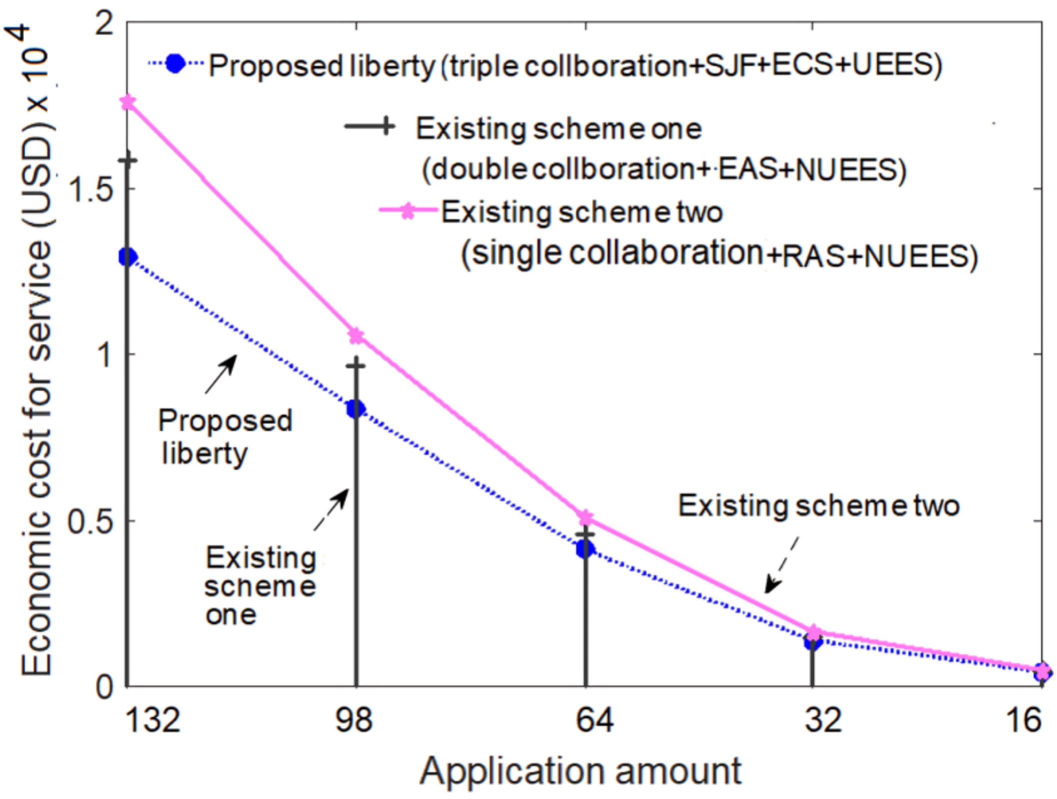


(a)

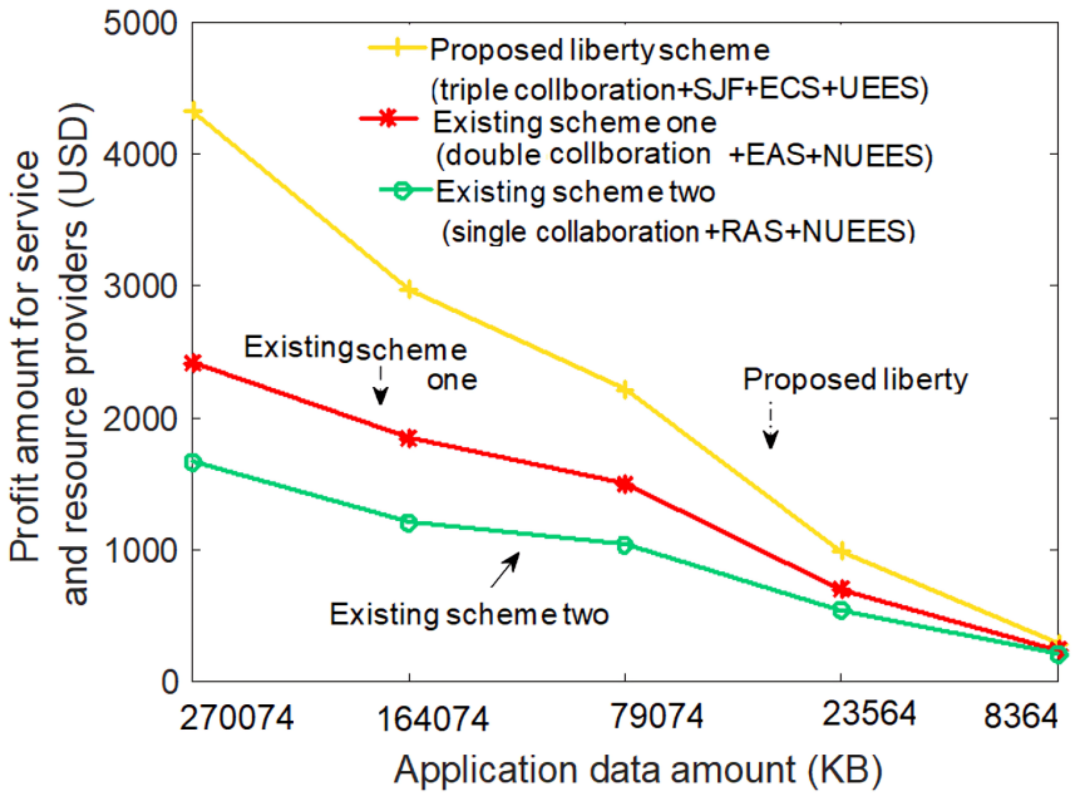


(b)

Fig. 4. Sucessful task completion ratio and network throughput

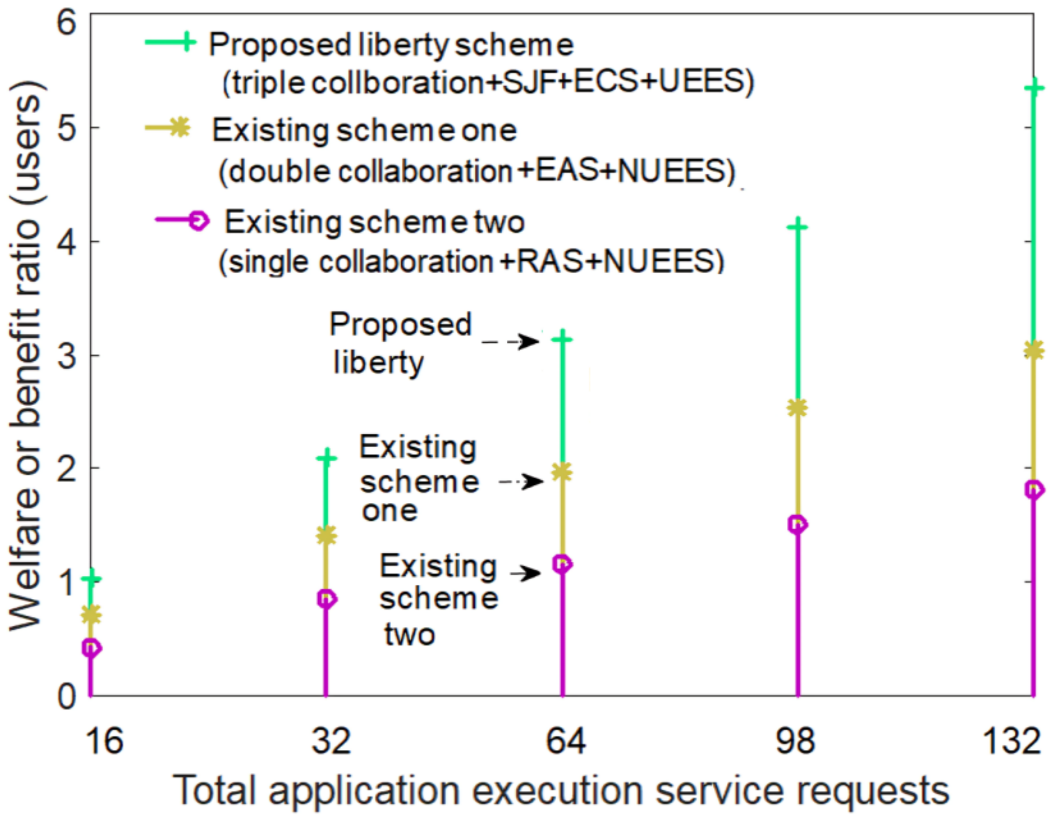


(a)

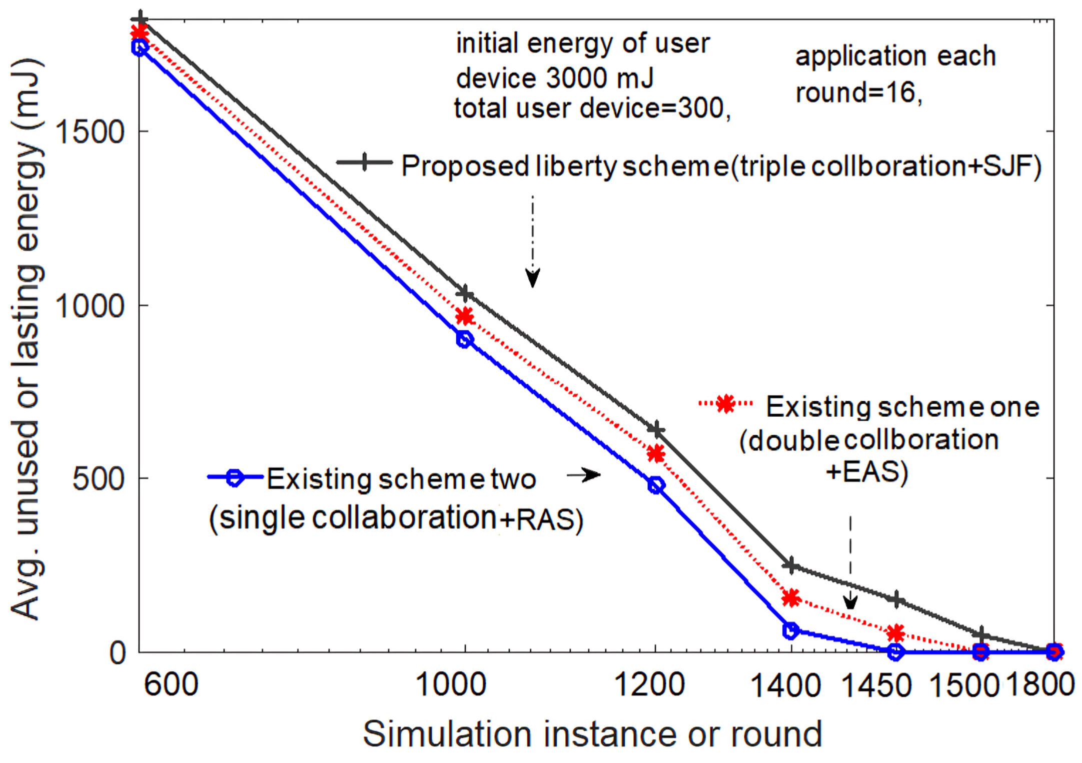


(b)

Fig. 5. Users economic cost and profit for service providers

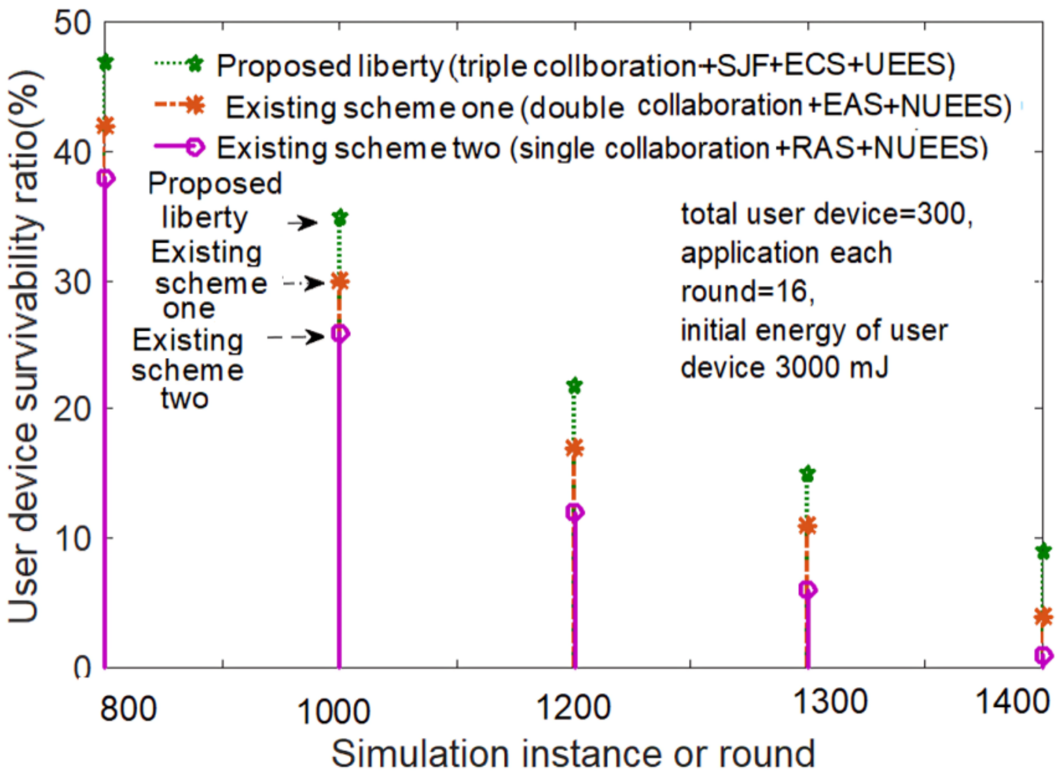


(a)

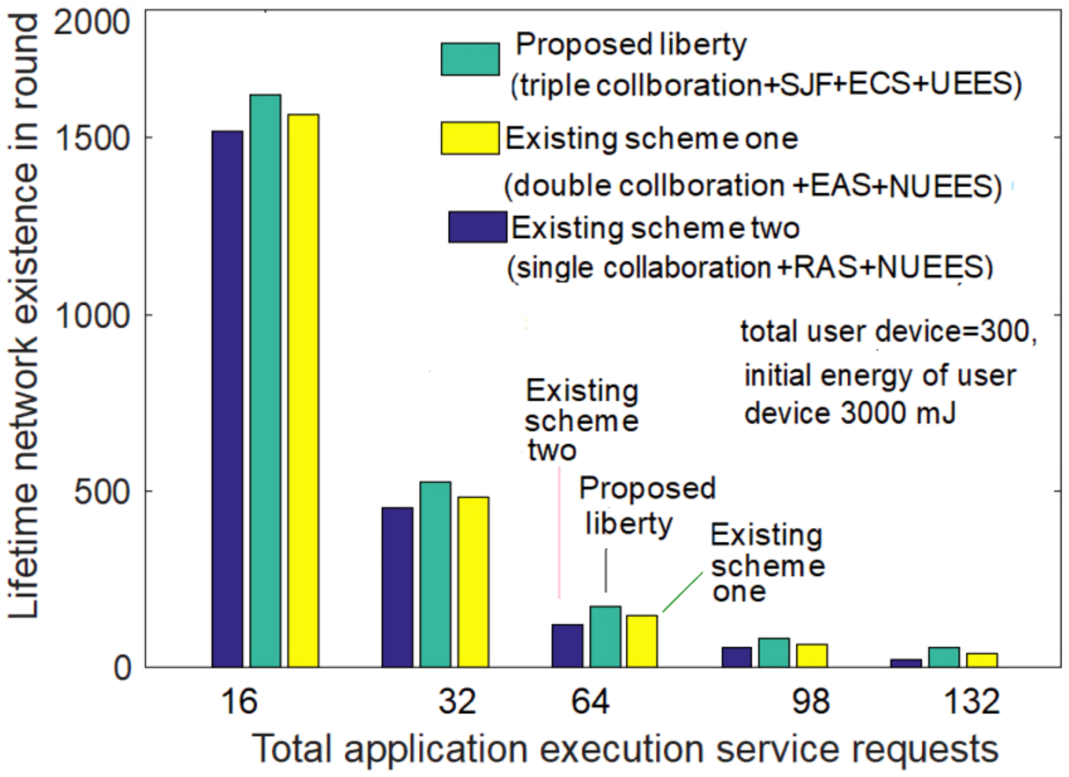


(b)

Fig. 6. Welfare ratio for users and average unused energy



(a)



(b)

Fig. 7. User device survivability ratio and lifetime of network existence

TABLE I
EXPERIMENTAL NOTATIONS AND VALUES

Parameters	Values/Units
z, $\Psi^k_{eb}, \Psi^k_{rcr}, \Psi^k_{rce}, \Psi^k_{asr}$, Ψ^k_{ase}, Ψ^k_{iasb}, Q^{ep}_{pd}, Q^{ep}_{tm}, Λ_{pd}, Λ_{dd}, J_{wr}, J_{or}, u_{or}, and u_{wr}, β^k_{pp}, β^k_{rwd}	Total application request, primary beacon message size (160 bits), network registration request size (160 bits), network registration response size (160 bits), application service registration request size (160 bits), application service registration response size (160 bits), initial time-slot assign schedule message size (192 bits), workload for user device (10 CPU cycles/bit), workload for task managers during elementary phase (50 CPU cycles/bit), workload processing speed for user/physical worker (1200 MHz) and task manager/MEC/digital worker (2.5-4.5 GHZ), wireless link rate (10-50 Gbps for THz link, 2-8 Gbps for mm wave link, 50-500 Mbps for microwave link), fiber link data exchange rate (20-50 Gbps), hop number (optical fiber link) transfer (1-5), hop number for wireless link based data transfer (1-5), propagation delay (ms, vary), resource access queuing and waiting latency (ms, vary).
$\Psi^k_{uarc}/\Psi^k_{ar}, \Psi^k_{tmrt}$, $\Psi^k_{rmrw}, \Psi^k_{otmr}$, Ψ^k_{otmp}, Q^{sp}_{pd}, Q^{sp}_{dd}, Q^{sp}_{cm}, Ψ^k_{urb}, and Ψ^k_{wrb}, q_{cim}, q^k_{tmp}, total number of user-owned, neighbor-owned, and service provider-owned digital workers at network edge, service provider-owned remote digital workers, total user devices	Data size regarding user app request (1 KB), resource status request message size (192 bits), resource status response message size (192 bits), resource schedule information request (160 bits), resource schedule information response message (192 bits), workload associated with user device (10 cpu cycles/bit), workload associated with task manager and digital worker device during second phase (10 cpu cycles/bit), task managers workload count for resource schedule and application execution timeslot assignment work (10 cpu cycles/bit), resource schedule and application execution timeslot assignment message size for user device (5 cpu cycles/bit) and worker device (50 cpu cycles/bit), workload required during internet connectivity work (50 cpu cycles/bit), workload for task manager during application processing instruction generation, 2, 2, 4, 4, 300.

TABLE II
EXPERIMENTAL NOTATIONS AND VALUES (CONT.)

Parameters	Values/Units
y^k_{et}, y^k_{er}, y^k_{pd}, y^k_{vc}, y^k_{ew}, ς^k_{bw}, ς^k_{ud}, ς^k_{ew}, ς^k_{uo}, ς^k_{sh}, $\varsigma^k_{sp}, \varsigma^k_{sbw}, \varsigma^k_{sud}$, $\varsigma^k_{sew}, \varsigma^k_{suo}$, ς^k_{ssh}, ς^k_{ssp}, work simulation network area, d^{vet}_{xmu}, Λ_{pm}, $x_{rr}, x_{aa}, x_{bn}, x_{th}$, and a_{cr}, Ψ^{vet}_{rdt}	Energy expenditure avg. cost (per ms) for user host devices concerning data transmission (.44W), data receive activity (.35W), physical device based work processing (.54W), digital device based work processing (.0008W), resource access waiting activity (.0005W), economic expenditure avg. cost (per ms) for bandwidth usage (.4 usd for ultra-high, .3 usd for medium high, and .2 usd for low time critical app), physical device resource usage (.3 usd for ultra-high, .25 usd for medium high, and .2 usd for low time critical application), waiting activity during resource access (.001 usd), economic expenditure avg. cost (per ms) concerning user owned VNF processing/digital work node resource usage (.3 usd for ultra-high, .25 usd for medium high, and .22 usd for low time critical application), sharing neighbor owned VNF processing/digital worker resource usage (.4 usd for ultra-high, .35 usd for medium high, and .32 usd for low time critical), service provider owned VNF/digital worker resource usage (.5 usd for ultra-high, .45 usd for medium high, and .40 usd for low time critical), economic expenditure avg. cost (per ms) for different service providers concerning bandwidth service delivery (.25/.2/.15), physical device usage service delivery (.2/.28/.18), resource access waiting (.002 usd), economic expenditure average cost (per ms) for service providers concerning user owned VNF/digital work node resource usage (.18/.16/.15 usd), sharing neighbor owned VNF/digital worker resource usage (.22/.20/.18 usd), service provider owned VNF/digital work node resource service delivery (.32/.30/.28 usd), 500m*500m, distance between vehicle and charging station (10-500 m), movement speed for EVs (5-100 m/s), required charge for vehicles (.25-.95), present charge availability at the vehicle (.1-.75), threshold for vehicles battery depletion (.1), capacity of vehilces battery (65 KWh), rate of charging for vehicles (40-50 KW), output size for EV energy transfer app (1-5 KB)

application service delivery latency and total application requests for the three compared schemes. Figure 3(a) shows that the large total application execution number has a higher mean service delivery latency than the small application execution number in all schemes tested. For different total application execution numbers, the proposed liberty scheme had a lower mean service delivery latency than the other two existing schemes. The notable reason is that the proposed liberty scheme only selects the best worker device and resources (i.e., communication and computation resource nodes with lower predicted service workload processing delay, communication delay, and waiting delay) for different applications (e.g., multiverse, metaverse, and non-metaverse) by checking not only the service provider resources but also the resources owned by the users and nearby collaborative users. In the proposed liberty scheme, three types of owner resources (user-owned, neighbor-owned, and service-provided-owned) are used to execute user applications via triple collaboration. Furthermore, the proposed liberty scheme (triple collaboration+SJF+ECS+UEES) combines the shortest deadline-based job scheduling (SJF) policy with the economy cost-saving (ECS) and user energy expenditure cost-saving (UEES) policies. It should be noted that in this study, service delivery delay latency includes a variety of delays such as communication delay, computation delay, request transfer delay, resource scheduling delay, and resource access waiting time.

TABLE III
EXPERIMENTAL NOTATIONS AND VALUES (CONT.)

Parameters	Values/Units
q^{ig}_{xpu}, q^{k}_{atn}/q^{k}_{nfw}, q^{k}_{nid}/q^{k}_{npdi}, q^{ig}_{dwp}, q^{ig}_{xpr}, q^{mt}_{xpu}, $q^{mt}_{atn}/q^{mt}_{nfw}/q^{mt}_{nid}/q^{mt}_{npdi}$, q^{mt}_{dwp}, q^{mt}_{xpr}, q^{vce}_{xpu}, $q^{vce}_{nfw}/q^{vce}_{nlb}/q^{vce}_{npdi}/q^{vce}_{nid}$, q^{vce}_{dwp}, q^{vce}_{xpp}, q^{vce}_{dwpp}, q^{vce}_{xpr}, q^{mh}_{xpu}, $q^{mh}_{atn}/q^{mh}_{nfw}/q^{mh}_{npdi}/q^{mh}_{nid}$, q^{mh}_{dwp}, q^{mh}_{xpuu}, q^{mh}_{xpr}, q^{mh}_{xpr}, q^{sa}_{xpu}, $q^{sa}_{nfw}/q^{sa}_{nlb}/q^{sa}_{nid}/q^{sa}_{npdi}$, q^{sa}_{dwp}, q^{sa}_{xpp}, q^{sa}_{dwpp}, q^{sa}_{xpr}, q^{hrp}_{xpu}, $q^{hrp}_{nfw}/q^{hrp}_{atn}/q^{hrp}_{nid}$, q^{hrp}_{dwp}, q^{hrp}_{xpuu}, q^{hrp}_{xpr}, q^{fs}_{xpu}, Ψ^{fs}_{di}, $q^{fs}_{nfw}/q^{fs}_{atn}/q^{fs}_{nid}$, $q^{fs}_{dwp}/q^{fs}_{dwpp}$, Ψ^{fs}_{rdt}, q^{fs}_{xpuu}	Workload for the XR device initial work for immersive gaming app (10 CPU cycles/bit), workload for NAT/firewall/IDS/DPI operation for immersive gaming (100 CPU cycles/bit), immersive gaming app workload for digital worker node (1K CPU cycles/bit), workload for the immersive gaming appication receiver device (10 CPU cycles/bit), XR device initial work instruction for virtual museum tour (100 CPU cycles/bit), workload tour (100 CPU cycles/bit), workload for the digital worker node during museum tour (1K CPU cycles/bit), virtual museum tour app final workload for the receiver device (10 CPU cycles/bit), workload for XR device initial virtual clothing task (10 CPU cycles/bit), workload for firewall/LB/DPI/IDS operation during virtual clothing experience task (100 CPU cycles/bit), workload at digital worker during virtual clothing work (500 CPU cycles/bit), workload for XR device based shopping data selection (10 CPU cycles/bit), digital worker workload for the second offloaded data processing during virtual clothing task (500 CPU cycles/bit), virtual clothing app workload for receiver device (10 CPU cycles/bit), workload amount for NAT/firewall/DPI/IDS operation for healthcare task (100 CPU cycles/bit), healthcare task workload for the offloaded data processing at digital worker (500 CPU cycles/bit), workload for doctors device (10 CPU cycles/bit), final workload for user device during healthcare task (10 CPU cycles/bit), initial workload for user device during avatar socialization (10 CPU cycles/bit), workload for firewall/LB/IDS/DPI operation for avatar socialization (100 CPU cycles/bit), digital worker workload for the meaverse socialization (500 CPU cycles/bit), workload for XR device based avatar socialization data selection (10 CPU cycles/bit), digital worker workload for secondary offloaded data processing (500 CPU cycles/bit), final workload for the receiver device for avatar socialization (10 CPU cycles/bit), initial workload for XR device during hologram based remote presence (10 CPU cycles/bit), VNF processing workload for firewall/NAT/IDS operation (100 CPU cycles/bit), workload for the MEC server during hologram based telepresence (500 CPU cycles/bit), decompress and decoding workload for the receiver (10 CPU cycles/bit), workload for the receiver device during remote telepresence data visualization (10 CPU cycles/bit), workload for XR device based data capture and local training work for FL based task (20 CPU cycles/bit), uploaded data size for FL based app (100-700 KB), workload for firewall/NAT/IDS operation for FL based app (100 CPU cycles/bit), workload for digital worker during initial offload and secondary operation during FL app (1K CPU cycles/bit), output data size for FL based surveillance app (100-700 KB), final result visualization workload for the receiver device during FL based app (10 CPU cycles/bit)

TABLE IV
EXPERIMENTAL NOTATIONS AND VALUES (CONT.)

Parameters	Values/Units
Ψ^{k}_{sip}, Ψ^{ig}_{di}, $\Psi^{k}_{tvpd}/\Psi^{k}_{stvd}$, Ψ^{ig}_{rdt}, Ψ^{mt}_{di}, $\Psi^{mt}_{tvpd}/\Psi^{mt}_{stvd}$, Ψ^{mt}_{rdt}, $\Psi^{vce}_{di}/\Psi^{vce}_{dii}$, $\Psi^{vce}_{tvpd}/\Psi^{vce}_{stvd}$, Ψ^{vce}_{rdt}, Ψ^{mh}_{di}, $\Psi^{mh}_{tvpd}/\Psi^{mh}_{stvd}$, Ψ^{mh}_{dii}, Ψ^{mh}_{rdt}, Ψ^{sa}_{di}, $\Psi^{sa}_{tvpd}/\Psi^{sa}_{stvd}$, Ψ^{sa}_{dii}, Ψ^{sa}_{rdt}, Ψ^{hrp}_{di}, $\Psi^{hrp}_{tvpd}/\Psi^{hrp}_{stvd}$, Ψ^{hrp}_{rdt}, Ψ^{fs}_{dii}, $\Psi^{fs}_{tvpd}/\Psi^{fs}_{stvd}$, Ψ^{bnas}_{di}, Ψ^{bnas}_{dii}, Ψ^{bnas}_{rdt}, $\Psi^{bnas}_{tvpd}/\Psi^{bnas}_{stvd}$,	Data size that includes instruction for workers (1 KB), immersive gaming app input data (1-7 MB), exchanged data between VNF node for gaming (1-7 MB), output data size for immersive gaming (1-7 MB), museum app input data (1-5 MB), exchanged data amount for museum tour app (1-5 MB), output data size for museum tour app (1-5 MB), offloaded data for virtual clothing (1-6 MB), exchanged data amount for virtual clothing (1-6 MB), output data size for virtual clothing app (1-6 MB), input data size for healthcare task (100-700 KB), exchanged data amount for healthcare app (100-700 KB), MEC processed data size for healthcare task (100-700 KB), final output data size for healthcare task (100-700 KB), initial input data size for avatar based socialization task (10-70 KB), exchanged data amount for VNF processing during avatar socialization app (10-70 KB), input data size for socialization app data (10-70 KB), output data size for avatar to avatar interaction experience (10-70 KB), input data size for remote telepresence app (1-5 MB), exchanged data amount for VNF processing during remote telepresence app (1-5 MB), output data size for remote presence app (1-5 MB), global deep learning model data for FL based surveillance app (1-5 MB), exchanged data amount for VNF processing during FL based app (1-5 MB), input data size for NFT and BC based visual art selling work (1-6 MB), data size that consists the buyer user XR device captured visual art buying data (1-6 MB), output data size that consists NFT trading finalization data (1-6 MB), VNF server exchanged data amount during NFT trading application (1-6 MB)
Ψ^{drs}_{di}, $\Psi^{drs}_{tvpd}/\Psi^{drs}_{stvd}$, Ψ^{drs}_{dii}, Ψ^{drs}_{rdt}, $\Psi^{ia}_{di}/\Psi^{ia}_{rdth}$, $\Psi^{ia}_{tvpd}/\Psi^{ia}_{stvd}$ and $\Psi^{av}_{tvpd}/\Psi^{av}_{stvd}$, Ψ^{ia}_{rdtc}, Ψ^{ia}_{rdtt}, $\Psi^{av}_{di}/\Psi^{av}_{rdt}$, Ψ^{tb}_{di}, $\Psi^{tb}_{tvpd}/\Psi^{tb}_{stvd}$, $\Psi^{tb}_{di}/\Psi^{tb}_{rdt}$, Ψ^{vc}_{di}, $\Psi^{vc}_{tvpd}/\Psi^{tb}_{stvd}$, Ψ^{vc}_{rdt}, $\beta^{vc}_{ew}/\beta^{vc}_{rw}$, $\Psi^{bcx}_{di}/\Psi^{bcx}_{rdt}$, $\Psi^{bcx}_{tvpd}/\Psi^{bcx}_{stvd}$, $\Psi^{vet}_{tvpd}/\Psi^{vet}_{stvd}$	Input data size for digital twin based robot selection (20-120 KB), exchanged data amount VNF processing for digital twin based task (20-120 KB), data size for robot captured data (20-120 KB), output data size for digital twin based task (20-120 KB), input data size for industrial automation (10-70 KB), exchanged data amount during VNF processing for industrial automation (10-70 KB) and autonomous vehicles application (100-700 KB), MEC procesed primary output data for industrial automation (10-70 KB), human worker device offloaded data size (10-70 KB), input/output data size for autonomous vehicle app (100-700 KB), input/output data size for traditional broadband app (10-70 KB), exchanged data amount during VNF processing for broadband app (10-70 KB), input data size for caching app (1 KB), exchanged data amount during VNF processing for video caching app (4-32 KB), output data size for video caching app (4-32 KB), communication delay for edge caching/remote server caching (ms, vary), input data size (10-70 KB) and output data size (10-70 KB) for human-brain sensor-wheel chair interface based app, exchanged data amount during VNF processing for human brain-computer-wheelchair interaction app (10-70 KB), exchanged data amount during EV energy transfer app VNF processing operation (1-5 KB)

TABLE V
EXPERIMENTAL NOTATIONS AND VALUES (CONT.)

Parameters	Values/Units
q_{bcr}^{bnas}/q_{bcrr}^{bnas}, q_{xpu}^{bnas}, q_{nfw}^{bnas}/q_{atn}^{bnas}/q_{nid}^{bnas}, q_{dwp}^{bnas}, q_{xpuu}^{bnas}, q_{dwpp}^{bnas}, q_{bcc}^{bnas}, Ψ_{dib}^{bnas}, q_{bvm}^{bnas}, q_{dwps}^{bnas}, q_{xpr}^{bnas}, q_{xpu}^{drs}, q_{nfw}^{drs}/q_{atn}^{drs}/q_{nid}^{drs}, q_{dwp}^{drs}, q_{xpuu}^{drs}, q_{dwop}^{drs}, q_{dwpp}^{drs}, q_{xpr}^{drs}, q_{xpu}^{ia}/q_{xpus}^{ia}, q_{nfw}^{ia}/q_{atn}^{ia}/q_{nid}^{ia}/q_{npdi}^{ia}, q_{dwp}^{ia}, q_{xpuc}^{ia}, q_{xpuh}^{ia}/q_{xpr}^{ia}, q_{xpu}^{av}, q_{nfw}^{av}/q_{atn}^{av}/q_{nid}^{av}/q_{npdi}^{av}, q_{dwp}^{av}, q_{xpr}^{av}, q_{xpu}^{tb}, q_{nfw}^{tb}/q_{atn}^{tb}/q_{nid}^{tb}/q_{npdi}^{tb}, q_{dwp}^{tb}, q_{xpu}^{vc}/q_{xpr}^{vc}, q_{nfw}^{vc}/q_{atn}^{vc}/q_{nid}^{vc}/q_{npdi}^{vc}, q_{dwp}^{vc}, q_{xpu}^{bcx}, q_{nfw}^{bcx}/q_{atn}^{bcx}/q_{nid}^{bcx}/q_{npdi}^{bcx}, q_{dwp}^{bcx}, q_{xpr}^{bcx}, q_{nfw}^{vet}/q_{atn}^{vet}/q_{nid}^{vet}/q_{npdi}^{vet}, q_{xpr}^{vet},	Workload for user/blockchain worker device during initial blockchain operation (100 CPU cycles/bit), workload for XR device based avatar data capture (10 CPU cycles/bit), workload for firewall/NAT/IDS operation regarding NFT and BC based art selling app (100 CPU cycles/bit), digital worker workload for during NFT based selling (500 CPU cycles/bit), workload for the buyer user (10 CPU cycles/bit), secondary workload for the digital worker node for art selling (500 CPU cycles/bit), workload for the block creation at digital worker (100 CPU cycles/bit), data size that consists the block creation data (1-6 MB), workload for the block verification at digital worker (100 CPU cycles/bit), workload for the ledger update by digital worker (50 CPU cycles/bit), workload for the receiver during NFT based selling (10 CPU cycles/bit), Workload for user device data capture for digital twin simulation task (10 CPU cycles/bit), workload for firewall/NAT/IDS/DPI operation (100 CPU cycles/bit), digital worker initial workload for digital twin simulation task (500 CPU cycles/bit), workload for robots own data capture (10 CPU cycles/bit), workload for digital twin based task processing (500 CPU cycles/bit), workload for digital worker during digital twin based task (500 CPU cycles/bit), workload for the receiver during digital twin based task (10 CPU cycles/bit), workload for video camera for data capture (10 CPU cycles/bit), VNF processing workload for firewall/NAT/IDS during industrial automation (100 CPU cycles/bit), initial workload for digital worker during industrial automation work (500 CPU cycles/bit), workload for cobot (10 CPU cycles/bit), workload for human worker/receiver device (10 CPU cycles/bit), workload for vehicles camera (10 CPU cycles/bit), workload for firewall/NAT/IDS/DPI operation for autonomous vehicle app (100 CPU cycles/bit), workload for digital worker during autonomous vehicle app (500 CPU cycles/bit), workload for the receiver device during autonomous vehicle application (10 CPU cycles/bit), workload for user device for personal data capture (10 CPU cycles/bit), VNF processing workload for firewall/NAT/IDS during traditional broadband app (100 CPU cycles/bit), workload for digital worker during traditional broadband app (500 CPU cycles/bit), workload for host user device/receiver device during caching application (10 CPU cycles/bit), workload for firewall/NAT/IDS/DPI operation for video caching application (100 CPU cycles/bit), workload for digital worker during video caching (500 CPU cycles/bit), workload for brain sensor based data capture operation/receiver device during human brain-computer-wheelchair interaction app (10 CPU cycles/bit), workload for firewall/NAT/IDS/DPI operation during human brain-computer-wheelchair interaction app (100 CPU cycles/bit), workload for digital worker during human brain-computer-wheelchair interaction app (500 CPU cycles/bit), workload for firewall/NAT/IDS/DPI operation during charging station to vehicles electric energy transfer app (100 CPU cycles/bit), workload for the receiver device during energy transfer app (10 CPU cycles/bit)

Figure 3(a) shows that the existing scheme one provides the second-best mean service delivery delay, while the existing scheme three provides the worst delay. The primary reason for this is that the existing scheme one only uses two types of resources (user-owned and service provider-owned) for application execution. The existing scheme one, which has a no-user energy expenditure saving policy (double collaboration+EAS+NUEES), allocates resources for application execution based on the earliest application request (EAS). The existing scheme two (single collaboration+RAS+NUEES) relies solely on service provider resources and does not include any user-owned or neighbor-owned resources. The existing scheme uses random approaches to application and resource scheduling. Thus, the existing scheme two provides the worst result in terms of mean service delivery delay. Figure 3(a) shows that when the total number of application execution requests is 132, the proposed liberty scheme, existing scheme one, and existing scheme two have mean service delivery delay latency of 16257 ms, 33705 ms, and 40794 ms, respectively.

Figure 3(b) depicts the change in users' energy expenditure for three different schemes as the number of application requests increases. Figure 3(b) shows that increasing the total number of application execution requests can significantly increase users' energy expenditure value in both existing and proposed liberty schemes. The reason for this type of outcome is that more application execution requires more energy consumption on user devices due to application workload processing, data transmission and reception, and resource access wait activities. The proposed liberty scheme requires minimal energy expenditure from users. This is because the proposed liberty scheme reduces application service delivery delays by selecting the best resources and using three types of collaborative resources. Users experience lower energy efficiency as service delivery delays are reduced. Furthermore, the proposed liberty scheme follows the user energy energy expenditure cost-saving policy. This policy allows users to go to sleep mode during other devices' time slots.They can get up right before their own time slot. The proposed liberty scheme reduces energy waste by preventing idle listening during other users' time slots.Users' energy expenditure costs are highest in existing scheme two and second best in existing scheme one. The primary reason is that existing schemes do not include a user energy-saving policy based on device sleep and wake-up policies. Another important factor is that the application service delivery delay latency is the highest and second highest in existing schemes two and one, respectively. This additional delay in application service delivery necessitates a higher level of energy expenditure in both current schemes. Figure 3(b) shows that when the total number of application execution requests is 98, users' energy expenditures for the proposed liberty scheme, existing scheme one, and existing scheme two are 10794 mJ, 12614 mJ, and 13550 mJ.

Figure 4(a) depicts the relationship between the three schemes' successful task completion ratios and the time limit for application execution. The proposed liberty scheme's successful task completion ratio is always higher than both existing schemes one and two, regardless of the size of the time limit (deadline for execution). It is clear that when the application execution time limit is set to a high value, the successful task completion ratio is high in all comparison schemes. In this paper, the successful task completion ratio is defined as the ratio of the total number of successful application executions meeting the time limit criteria to the total number of application requests. The proposed liberty scheme can select the best resources (communication, application workload processing device, caching node) with a lower predicted application execution or service delivery delay by implementing the triple collaboration policy (e.g., utilizing host user-owned, neighbor-owned, and service provider-owned resources), the SJF scheduling policy, and economic cost savings. The existing scheme one (with a double collaboration policy and an earliest application arrival first-based resource scheduling policy) has the second-best communication delay, resource access waiting time, and application work processing delay. Thus, the existing scheme two can provide the lowest success task completion ratio for various application execution time limits. The existing scheme two (with a single collaboration policy and a random-based resource scheduling policy) has the highest communication delay, resource access waiting delay, and application workload processing delay. Figure 4(a) shows that when the application execution time limit is 39000 ms and the application arrival number is 132, the proposed liberty scheme, existing scheme one, and existing scheme two have successful task completion ratios of 75 percent, 66 percent, and 60 percent, respectively.

Figure 4(b) depicts the effect of application data size on network throughput value. The network throughput can be calculated as the ratio of the amount of data exchanged to the total time it takes to execute an application. As the number of executed applications grows, the throughput of the three compared schemes gradually increases (see figure 4(b)). The proposed liberty scheme selects the best resources (e.g., cloud and network) for application execution through triple collaboration (i.e., use of user-owned, neighbor-owned, and service provider-owned resources) and by ensuring that they have lower communication, application work processing delay, and resource access waiting delay. As a result, the proposed liberty scheme achieves higher network throughput than others. The existing scheme one has the second-highest application execution timespan delay, while the existing scheme two has the highest application execution timespan value. As a result, the network throughput value is the second-lowest in schemes one and lowest in previous scheme two, respectively. Figure 4(b) shows that for 23564 KB of executed application data, the proposed liberty scheme, existing scheme one, and existing scheme two have network throughput values of 15.81 Mbps, 7.45 Mbps, and 6.3 Mbps, respectively.

Figure 5(a) reflects the user's economic expenditure costs for various schemes by varying the application execution amounts. As application executions increase, users incur economic costs for service execution and resource usage in three areas. Compared schemes show an upward trend. The proposed liberty schemes have a much lower economic cost than the existing schemes one and two. The primary reason for this is that the proposed liberty scheme reduces resource utilization time by selecting the best resource with the lowest predicted service delivery delay latency. Economic costs associated with application service execution include coordination, communication, computation, and resource caching. It should be noted that the proposed liberty scheme selects the best resources with the least communication, computation, caching, and resource access waiting delay by examining three different types of resources (host user-owned, collaborative neighbor-owned, and service provider-owned). The resource utilization time (i.e., application service delivery delay) is greatest in existing scheme two and second best in existing scheme one. Thus, existing scheme two has the highest economic expenditure cost for users, while existing scheme one has the second lowest. Figure 5(a) shows that with 98 executed application data, the user's economic expenditure cost value for the proposed liberty scheme, existing scheme one, and existing scheme two is 8364 USD, 9647 USD, and 10576 USD, respectively.

Figure 5(b) depicts changes in service provider profit as the number of executed application data requests increases. The results in Figure 5(b) show that for both large and small data amounts, the proposed liberty scheme can maximize profit for service providers. The primary reason for this is that the proposed liberty scheme can reduce resource access waiting time and utilization time by selecting the best available resources and scheduling policy. The proposed liberty scheme also has the highest number of applications completed by the deadline when compared to others. The existing scheme one has the second-highest resource access waiting time during application execution. The existing scheme 2 has the longest resource access waiting delay during application execution. Because of increased overhead, fewer application executions, and an inefficient resource scheduling policy, the service provider's profit is lowest in existing scheme two and second lowest in existing scheme one. Figure 5(b) shows that for an executed application data amount of 270074 KB, the service provider profit value for the proposed liberty scheme, existing scheme one, and existing scheme two is 4330 USD, 2420 USD, and 1670 USD, respectively.

Figure 6(a) shows the welfare ratio for users under various numbers of application execution requests and three compared schemes. The results show that the welfare ratio for users increases as the application execution number increases in all three schemes. In this study, the welfare ratio is calculated by adding the service delivery delay gain ratio, the energy expenditure gain ratio, and the economic cost gain ratio. Figure 6(a) shows that the proposed liberty scheme provides the best welfare or benefit ratio for users by having the highest service delivery delay gain ratio, energy expenditure gain ratio, and economic cost gain ratio. Similarly, the existing scheme two provides the lowest welfare or benefit ratio to users due to the lowest service delivery delay gain ratio, energy expenditure gain ratio, and economic cost gain ratio. The existing scheme one provides the second-best welfare or benefit ratio for users due to its second-best service delivery delay gain ratio, energy expenditure gain ratio, and economic cost gain ratio. For example, in Figure 6(a), when the executed application amount is 64, the user welfare ratios for the proposed liberty scheme, existing scheme one, and existing scheme two are 3.15, 1.96, and 1.16, respectively.

Figure 6(b) shows that the average unused (leftover) energy of users decreases as the simulation round or instance number increases in both the proposed liberty scheme and the existing schemes. It is worth noting that the energy cost or expenditure value of user devices for application execution increases with the number of simulation rounds. However, due to lower resource usage time and lower energy expenditure per round, combined with the user energy saving policy, the proposed liberty scheme has the highest average unused (leftover) energy amount on the user's device. Whereas, due to incorrect resource selection, higher resource usage time per application, and a lack of an energy-saving policy, the average unused energy value is lowest in existing scheme two and second lowest in existing scheme one. This means that energy waste is minimal in our proposed liberty scheme compared with others. It can be seen in figure 6(b) that when the simulation instance amount is 1400, the total user device is 300, the initial energy of the user device is 3000 mJ, and the application execution number per round is 16, the average unused energy values for the proposed liberty scheme, existing scheme one, and existing scheme two are 250 mJ, 158 mJ, and 64 mJ, respectively.

Figure 7(a) depicts the relationship between the user survivability ratio and the simulation instance or round in three different schemes. The figure shows that increasing the number of simulation instances or rounds shortens.The user-device survival ratio. The user survivability ratio is calculated by dividing the number of alive user devices with sufficient energy after the simulation round by the total number of users at the start of the application execution process. As the number of simulation rounds increases, the number of active users decreases due to a lack of remaining energy in each compared scheme.

The proposed liberty scheme has the highest number of alive user device numbers with sufficient remaining energy due to lower energy expenditure at each round for varying application execution numbers. The number of alive user devices is the lowest in the current scheme two due to the higher amount of energy expended at each round for different numbers of application executions. The number of alive user devices is second best in the existing scheme one due to the second best energy expenditure at each round for varying application execution numbers. As the number of simulation rounds increases, the number of active users decreases due to a lack of remaining energy in each compared scheme. It can be visualized in figure 7(a) that when the simulation instance amount is 1300, the total user device is 300, the initial energy of the user device is 3000 mJ, and the application execution number per round is 16, the user survivability ratios for the proposed liberty scheme, existing scheme one, and existing scheme two are 14 percent, 9 percent, and 4 percent, respectively.

As illustrated in Figure 7(b), the network lifetime under three different schemes (both proposed and existing) decreases as the number of total application executions increases. Increasing the number of networked applications can reduce their effectiveness.the user device's remaining energy. The network lifetime can be calculated by dividing the total available energy of users by the total energy expenditure of users in each round for various application executions. The proposed liberty scheme (with triple collaboration, the shortest application deadline, first-based scheduling, and an economic and energy-saving policy) provides the best network lifetime value due to the low amount of energy spent at each simulation instance when compared to others. The existing scheme two (with single collaboration, random resource scheduling, and no energy-saving policy) has the lowest network lifetime value due to higher energy expenditure at each simulation instance than the others. The existing scheme one (with double collaboration, earliest arrival-based resource scheduling, and no energy-saving policy) provides the second-highest network lifetime value due to the second-best amount of energy expenditure at each simulation instance when compared to others. Figure 7(b) shows that when the application amount is 132 per round, the total user device is 300, and the user device's initial energy is 3000 mJ, the network lifetime for the proposed liberty scheme, existing scheme one, and existing scheme two is 54, 42, and 36 simulation rounds, respectively. The results show that the proposed liberty scheme outperforms the existing scheme for multiverse, metaverse, and non-metaverse-based 6G application execution.

A. *Comprehensive comparison results*

Tables VI and VII show the comprehensive evaluation results of our proposed liberty scheme (triple collaboration+SJF+ECS+UEES), the compared scheme (double collaboration+EAS+NUEES) one (e.g., [38], [70]), the compared scheme (single collaboration+RAS+NUEES) two (e.g., [37], [137]), and the compared scheme (double collaboration+MCDS+NUEES) three (e.g., [58], [78]), and existing scheme (single collaboration+GFS+NUEES) four (e.g., [21], [139]). We compared the results of the proposed scheme to existing schemes in terms of user economic cost, energy expenditure, user welfare ratio, mean service delivery latency, service provider profit, task completion ratio, and network throughput. Table VI shows that when the application completion number is 132, the proposed liberty scheme has lower mean service delivery latency, user economic cost, and energy expenditure than the other schemes tested. Table VII also shows that, for app number 132, the proposed liberty scheme outperforms the existing schemes in terms of service provider profit, network throughput, task completion ratio, and network lifetime (simulation round persistance) values. The compared scheme (double collaboration+EAS+NUEES) one ranks second, while the compared scheme (single collaboration+RAS+NUEES) two ranks fifth across all performance metrics. The compared scheme (double collaboration+MCDS+NUEES) three achieves third position among all compared methods. The previous scheme (single collaboration+GFS+NUEES) four secures fouth position among all schemes.

The proposed liberty scheme is effective because it employs the shortest job or lowest deadline-based scheduling technique for multiple application executions. However, by leveraging the triple collaboration feature, the proposed liberty scheme is dependent on the use of host user-owned resources, neighbor-owned resources, and service provider resources for various application executions. The proposed liberty scheme also uses minimum predicted communication, workload processing, and waiting delays to select the best resources.

The compared scheme one (double collaboration+EAS+NUEES) achieves the second-best results

TABLE VI
COMPREHENSIVE RESULTS AND EVALUATION FOR APP=132

Scheme Name	User economic cost	Energy expenditure of users	Welfare ratio for users	Mean service delivery latency
Proposed liberty scheme (triple collaboration+SJF+ECS+UEES)	12928 USD	16461 mW	5.36	16257 ms
Existing scheme (double collaboration+EAS+NUEES) one ([70], [38])	15860 USD	21611 mW	3.05	33705 ms
Existing scheme (single collaboration+RAS+NUEES) two ([37], [137])	17612 USD	23811 mW	1.81	40794 ms
Previous scheme (double collaboration+MCDS+NUEES) three ([58], [78])	16882 USD	22740 mW	2.42	37025 ms
Previous scheme (single collaboration+GFS+NUEES) four ([21], [139])	17220 USD	23290 mW	2.15	38660 ms

TABLE VII
COMPREHENSIVE COMPARISON FOR APP=132

Scheme Name	Providers profit	Task completion ratio	Network throughput	Network lifetime (round)
Proposed liberty scheme (triple collaboration+SJF+ECS+UEES)	4330 USD	100%	16.81 Mbps	54
Existing scheme (double collaboration+EAS+NUEES) one ([70], [38])	2420 USD	91.17%	8.12 Mbps	42
Existing scheme (single collaboration+RAS+NUEES) two ([37], [137])	1670 USD	80.88%	6.98 mbps	36
Previous scheme (double collaboration+MCDS+NUEES) three ([58], [78])	2050 USD	85.6%	7.6 Mbps	39
Previous scheme (single collaboration+GFS+NUEES) four ([21], [139])	1844 USD	83.2%	7.28 Mbps	37

because it only uses user and service provider-owned resources for all application executions. Furthermore, the optimal resource for each application execution is chosen based on the maximum CPU power. They also used the earliest arrival-based scheduling (EAS) technique to schedule multiple jobs, as well as the non-user energy expenditure cost-saving process (NUEES). The existing scheme two (single collaboration+RAS+NUEES) selects the best resources by combining random applications with the worker scheduling process (RAS). As a result, resource selection, queuing, and waiting delay are all largest in existing scheme two. The existing scheme three (double collaboration+MCDS+NUEES) relies on minimum communication delay latency (MCDS) based nearby suitable resource utilization for assigned task execution. This scheme can also utilize both user and service provider resources (double collaboration). There is no user energy cost saving policy is this existing scheme three (NUEES). Thus, the existing scheme three suffers from third best service execution latency. The resource selection, queuing, and waiting delays are the third best in existing scheme three (double collaboration+MCDS+NUEES). The existing scheme four (single collaboration+GFS+NUEES) depends on only service provider resources. They utilizes greedy scheduling technique (GFS) for resource selection and first job request arrival based appliication scheduling technique with no user energy cost saving policy (NUEES). The resource selection, queuing, and waiting delays are the fourth best in existing scheme four.

The proposed liberty scheme has the shortest resource selection, queuing, and waiting delays. Table VI displays the mean service delivery latency for the proposed liberty scheme, existing scheme one, existing scheme two, existing scheme three and existing scheme four, which are 16257 ms, 33705 ms, 40794 ms, and 37025 ms, and 38660 ms, respectively. Table VII shows that our proposed liberty scheme has the highest network lifetime value (for 132 application executions) among the compared schemes. The network lifetime value of the proposed liberty scheme, existing schemes one, two, three, and four are 54, 42, 36, 39, and 37, respectively.

Tables VI and VII clearly depict that for the execution of 132 applications, the proposed liberty scheme achieves 22.67% more economic cost gain, 31.28% more energy expenditure cost gain, 42.91% more welfare ratio gain, 44.11% more providers profit gain, 8.83% more task completion success gain, and 51.75% more network throughput gain than the compared scheme (double collaboration+EAS+NUEES) one. Tables VI and VII also hint that for the execution of 132 applications, the proposed liberty scheme shows 36.23% economic cost gain, 44.65% energy expenditure cost gain, 66.23% welfare ratio gain, 61.43% providers profit gain, 19.12% task completion success gain, and 58.48% network throughput gain than the compared scheme two (single collaboration+RAS+NUEES). Tables VI and VII also notify that for the execution of 132 applications, the proposed liberty scheme shows 30.58% economic cost gain, 38.14% energy expenditure cost gain, 54.85% welfare ratio gain, 52.65% providers profit gain, 14.4% task completion success gain, and 54.78% network throughput gain than the compared scheme three (double collaboration+MCDS+NUEES). Tables VI and VII depict that for the 132 applications, the proposed liberty scheme gives 33.19% economic cost gain, 41.48% energy expenditure cost gain, 59.88% welfare ratio gain, 57.41% providers profit gain, 16.8% task completion success gain, and 56.69% network throughput gain than the compared scheme four (single collaboration+GFS+NUEES).

Thus, based on the findings, it can be concluded that the proposed liberty scheme is best suited for resource management in metaverse and multiverse-based application execution over 6G-enabled networks.

VI. Conclusion

This article describes a resource management scheme for multiverse, metaverse, and non-metaverse applications over 6G edge computing networks that considers economy, time criticality, three-fold collaboration (using different ownership-based resources), minimum service delivery delay latency, network function virtualization, and minimum energy cost. The proposed scheme integrates a resource and application scheduling scheme (based on shortest job first) along with a selected worker management scheme by taking different 6G and non-6G application requests, user, neighbor, and service provider-ownership-based digital worker resources (e.g., MEC server, blockchain node, caching server, digital twin, federated learning server), communication links, different physical worker resources, different application workloads, and application deadlines. In order to show the working sequence of various stages related to multiverse, metaverse, and non-metaverse application execution by considering the elementary network formation phase, information collection and resource mapping phase, and application work completion phase, Additionally, by accounting for various communication link types, user devices and physical workers, and different ownership-based digital workers (e.g., blockchain, MEC servers, FL servers, caching servers), this paper offers a novel network model for 6G edge computing services. To compute the performance of the proposed liberty scheme, this article provides an analytical or numerical model that includes the mean application service delivery delay, economic cost for users, successful task completion ratio, energy expenditure of users, throughput value, total benefit for service providers, user total benefit ratio, unused energy value, lifetime of the network, and survivability ratio of users. In order to highlight the advantages of the proposed liberty scheme, this article compares the simulation results of the proposed three-fold collboration-based liberty scheme with the existing scheme one (with no user energy saving policy, double collboration with user and service provider resource usage, earliest application arrival-based resource scheduling), and the existing scheme two (single collboration with only service provider resource usage, random scheduling-based resources, and no user energy saving policy). The experimental and performance comparison results illustrate that the suggested new liberty scheme delivers 76.98% mean app service delivery gain, 17.13% users energy expenditure gain, and 15.33% economic cost gain of users over the existing scheme one policy (double collaboration+EAS+NUEES) for the application work completion number of ninety-eight. The performance comparison results also visualize that the liberty scheme provides 115% mean app service delivery gain, 25.53% users energy expenditure gain, and 26.44% economic cost gain of users over the existing scheme two policy (single collaboration+RAS+NUEES) for the application work completion number of 98. The future research challenges and future extensions of this work may include game theory-based auction mechanism design for network resource selling, deep reinforcement learning-based routing paths, congested paths, best digital worker selection, service provider service delivery strategy selection, federated learning-based security threat detection, and network failure identification prediction for different multiverse, metaverse, and non-metaverse-based application execution over 6G edge computing networks.

References

[1] S. Dhelim et al., "Edge-enabled Metaverse: The Convergence of Metaverse and Mobile Edge Computing," *submitted to IEEE IoT journal,* pp. 1-8, 2022, https://doi.org/10.48550/arXiv.2205.02764.

[2] J. Kastrenakes and A. Heath, "Facebook is spending at least USD 10 billion this year on its metaverse division," *https://www.theverge.com/2021/10/25/22745381/facebook-reality-labs-10-billion-metaverse,* October 2021.

[3] D. Kalogeropoulos and A. Sharma ,"What Microsoft's Big Deal Means for Metaverse Investors," *https://www.fool.com/investing/2022/01/28/what-microsofts-big-deal-means-for-metaverse-inves/,* January 2022.

[4] M. Slovick, "The AR-VR age has begun in health care," *https://www.cta.tech/ Resources/i3-Magazine/i3-Issues/2020/November-December/ The-AR-VR-Age-has-Begun-in-Health-Care,* November 2020.

[5] S. Munde, "Augmented and Virtual Reality in Education Market Overview Source: https://www.marketresearchfuture.com/reports/ar-vr-in-education-market-10834," *https://www.marketresearchfuture.com/reports/ar-vr-in-education-market-10834,* July 2024.

[6] N. Pereira et al., "Arena: The augmented reality edge networking architecture," *IEEE International Symposium on Mixed and Augmented Reality (ISMAR),* pp. 479–488, 2021.

[7] K. Li et al., "When Internet of Things Meets Metaverse: Convergence of Physical and Cyber Worlds," *IEEE Internet of Things Journal,* vol. 10, no. 5, pp. 4148-4173, 1 March, 2023, doi: 10.1109/JIOT.2022.3232845.

[8] J. Joshua, "Information bodies: Computational anxiety in neal stephenson's snow crash," *Interdisciplinary Literary Studies,* vol. 19, no. 1, pp. 17–47, 2017.

[9] M. Xu et al., "A full dive into realizing the edgeenabled metaverse: Visions, enabling technologies, and challenges," *IEEE Communications Surveys and Tutorial,* vol. 25, no. 1, pp. 656-700, 2022.

[10] M. Ali et al., "Metaverse Communications, Networking, Security, and Applications: Research Issues, State-of-the-Art, and Future Directions," *in arXiv preprint,* pp. 1-38, 2022, https://doi.org/10.48550/arXiv.2212.13993.

[11] H. Duan et al., "Metaverse for social good: A university campus prototype," *29th ACM International Conference on Multimedia* , pp. 153–161, 2021.

[12] CoinMarketCap,"Top Metaverse Tokens by Market Capitalization," *https://coinmarketcap.com/tr/view/metaverse/* , May 2024.

[13] Young Professionals blog by IEEE, "Metaverse and Multiverse: The Real Sense of AR, VR, MR, XR and IR," *https://yp.ieee.org/blog/2022/04/13/metaverse-and-multiverse-the-real-sense-of-ar-vr-mr-xr-and-ir/* , April 2022.

[14] Optimize blog, "Metaverse versus multiverse key differences and comparison," *https://optymize.io/blog/metaverse-vs-multiverse-key-differences-and-comparison/* , November 2023.

[15] G. Weston, "Metaverse versus Multiverse," *https://101blockchains.com/metaverse-vs-multiverse/* , December 2021.

[16] J. D. N. Dionisio et al., "3d virtual worlds and the metaverse: Current status and future possibilities," *ACM Computing Surveys (CSUR),* vol. 45, no. 3, pp. 1–38, 2023.

[17] C. W. Thompson et al., "Next-generation virtual worlds: architecture, status, and directions," *IEEE Internet Computing,* vol. 15, no. 1, pp. 60–65, 2010.

[18] N. Abdenacer et al., "A Novel Framework for Mobile Edge Computing By Optimizing Task Offloading," *IEEE Internet of Things Journal,* vol. 8, no. 16, pp. 13065-13076, 2021.

[19] L.-H. Lee et al., "All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda," *arXiv preprint,* arXiv preprint arXiv:2110.05352, pp. 1-10, 2021.

[20] X.-Q. Pham et al., "Partial computation offloading in parked vehicle-assisted multi-access edge computing: A game-theoretic approach," *IEEE Transactions on Vehicular Technology,* vol. 71, no. 9, pp. 10220-10225, 2022.

[21] Ibrahim Aliyu et al., "A Dynamic Partial Computation Offloading for the Metaverse in In-Network Computing," *arXiV preprint,* pp. 1-14, 2023.

[22] K. Guo et al., "Joint computation offloading and bandwidth assignment in cloud-assisted edge computing," *IEEE Transactions on Cloud Computing,* vol. 10, no. 1, pp. 451-460, 2019.

[23] K. Poularakis et al., "Service placement and request routing in MEC networks with storage, computation, and communication constraints," *IEEE/ACM Transactions on Networking,* vol. 28, no. 3, pp. 1047-1060, 2020.

[24] J. Chen et al., "Non-cooperative game algorithms for computation offloading in mobile edge computing environments," *Journal of Parallel and Distributed Computing,* vol. 172, pp. 18-31, 2023.

[25] W. Yu et al., "Asynchronous Hybrid Reinforcement Learning for Latency and Reliability Optimization in the Metaverse over Wireless Communications," *IEEE Journal on Selected Areas in Communications (JSAC),* vol. 41, no. 07, pp. 2138-2157, 2023.

[26] Q. Yang et al., "Fusing blockchain and ai with metaverse: A survey," *IEEE Open Journal of the Computer Society,* vol. 3, pp. 122-136, 2022.

[27] H. Ning et al., "A survey on metaverse: the state-of-the-art, technologies, applications, and challenges," *IEEE Internet of Things Journal,* vol. 10, no. 16, pp. 14671-14688, 2023.

[28] T. Huynh-The et al., "Artificial intelligence for the metaverse: A survey," *Engineering Applications of Artificial Intelligence,* vol. 117, pp. 1-10, 2023.

[29] K. Y. Lam et al., "Human-avatar interaction in metaverse: Framework for full-body interaction," *in Proceedings of the 4th ACM International Conference on Multimedia in Asia,* Tokyo, Japan, pp. 1–7, 2022.

[30] L. U. Khan et al., "Metaverse for wireless systems: Architecture, advances, standardization, and open challenges," *arXiv preprint,* arXiv:2301.11441, pp. 1-10, 2023.

[31] J. Wang et al., "Wireless metaverse avatar construction," *arXiv preprint,* arXiv:2211.12720, pp. 1-10, 2022.

[32] H. Moudoud et al., "Federated Learning Meets Blockchain to Secure the Metaverse," *International Wireless Communications and Mobile Computing (IWCMC) conference,* Marrakesh, Morocco, pp. 339-344, 2023.

[33] T. H. The et al., "Blockchain for the metaverse: A review," *Future Generation Computer Systems,* Vol. 143, pp. 401-419, 2023.

[34] M. Xu et al., "A full dive into realizing the edge-enabled metaverse: Visions, enabling technologies, and challenges," *IEEE Communications Surveys and Tutorials,* vol. 25, no. 1, pp. 656-700, 2023.

[35] S.-M. Park et al., "A metaverse: taxonomy, components, applications, and open challenges," *IEEE Access,* vol. 10, pp. 4209-4251, 2022.

[36] Y. Wang et al., "A survey on metaverse: Fundamentals, security, and privacy," *IEEE Communications Surveys and Tutorials,* vol. 25, no. 01, pp. 319-352, 2023.

[37] S. Zhang et al., "Towards Green Metaverse Networking: Technologies, Advancements and Future Directions," *IEEE Network,* vol. xx, no. xx, pp. 1-10, 2023.

[38] Wei Yang Bryan Lim et al., "Realizing the metaverse with edge intelligence: A match made in heaven," *IEEE Wireless Communications,* vol. 30, no. 4, pp. 64-71, 2023.

[39] B. Siniarski et al., "Need of 6G for the Metaverse Realization," *IEEE Access,* vol. 4, 2023, pp. 1-31.

[40] L. Chang et al., "6G-Enabled Edge AI for Metaverse: Challenges, Methods, and Future Research Directions," *Journal of Communications and Information Networks,* vol. 7, no. 2, pp. 1-15, 2022.

[41] W. Saad et al., "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network,* vol. 34, no. 3, pp. 134–142, 2019.

[42] W. Jiang et al., "The Road Towards 6G: A Comprehensive Survey," *IEEE Open Journal of the Communications Society,* vol. 2, pp. 334-366, 2021.

[43] S. Karunarathna et al., "The Role of Network Slicing and Edge Computing in the Metaverse Realization," *in IEEE Access,* vol. 11, pp. 25502-25530, 2023.

[44] T. Umagiliya et al., "Network slicing strategies for smart industry applications," *in Proc. IEEE Conf. Standards Commun. Netw. (CSCN),* pp. 30–35, 2021

[45] A. Kaloxylos et al., "A survey and an analysis of network slicing in 5G networks," *IEEE Commun. Standards Mag.,* vol. 2, no. 1, pp. 60–65, 2018.

[46] S. Yang et al., "Delay-Sensitive and Availability-Aware Virtual Network Function Scheduling for NFV," *in IEEE Transactions on Services Computing,* vol. 15, no. 1, pp. 188-201, 2022.

[47] European Telecommunications Standards Institute, "ETSI Publishes First Specifications for Network Functions Virtualisation," *http://www.etsi.org/news-events/news/ 700-2013-10-etsi-publishes-first-nfv-specifications,* October 2013.

[48] A. M. Medhat et al., "Service function chaining in next generation networks: State of the art and research challenges," *IEEE Communications Magazine,* vol. 55, no. 2, pp. 216–223, 2017.

[49] D. C. Verma et al., "Service level agreements on IP networks," *Proceedings of the IEEE,* vol. 92, no. 9, pp. 1382–1388, 2004.

[50] V. Eramo et al., "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *in IEEE/ACM Transactions on Networking,* vol. 25, no. 4, pp. 2008-2025, 2017.

[51] W. Ma et al., "Traffic aware placement of interdependent NFV middleboxes", *in IEEE INFOCOM,* Atlanta, GA, USA, 2017, pp. 1-9.

[52] C. Pham et al., "Traffic aware and energy-efficient vNF placement for service chaining: Joint sampling and matching approach," *IEEE Transactions on Services Computing,* vol. 13, no. 1, pp. 172-185, 2020.

[53] J. Zhang et al., "On the theory of function placement and chaining for network function virtualization," *in Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing,* pp. 91–100, 2018.

[54] S. K. Taskou et al., "Energy and Cost Efficient Resource Allocation for Blockchain-Enabled NFV," *in IEEE Transactions on Services Computing,* vol. 15, no. 4, pp. 2328-2341, 2022.

[55] H. Li et al., "MSV: An Algorithm for Coordinated Resource Allocation in Network Function Virtualization," *in IEEE Access,* vol. 6, pp. 76876-76888, 2018.

[56] G. Sun et al., "Online Parallelized Service Function Chain Orchestration in Data Center Networks," *in IEEE Access,* vol. 7, pp. 100147-100161, 2019.

[57] H. Hawilo et al., "Network Function Virtualization-Aware Orchestrator for Service Function Chaining Placement in the Cloud," *in IEEE Journal on Selected Areas in Communications,* vol. 37, no. 3, pp. 643-655, 2019.

[58] G. Sun et al., "Low-latency and Resource-efficient Service Function Chaining Orchestration in Network Function Virtualization," *in IEEE IoT Journal,* vol. 7, no. 7, pp. 5760-5772, 2020.

[59] H. Hantouti et al., "Service Function Chaining in 5G and Beyond Networks: Challenges and Open Research Issues," *in IEEE Network,* vol. 34, no. 4, pp. 320-327, 2020.

[60] R. Souza et al., "NFV Data Centers: A Systematic Review," *in IEEE Access,* vol. 8, pp. 51713-51735, 2020.

[61] E. Amiri et al., "Deep Reinforcement Learning for Robust VNF Reconfigurations in O-RAN," *IEEE Transactions on Network and Service Management,* vol. 21, no. 1, pp. 1115-1128, 2024.

[62] H. Cao et al., "Resource-Ability Assisted Service Function Chain Embedding and Scheduling for 6G Networks With Virtualization," *in IEEE Transactions on Vehicular Technology,* vol. 70, no. 4, pp. 3846-3859, 2021.

[63] L. Dong et al., "On Application-Aware and On-Demand Service Composition in Heterogenous NFV Environments," *in IEEE Global Communications Conference (GLOBECOM),* Waikoloa, HI, USA, 2019, pp. 1-6.

[64] L. U. Khan et al., "Network Virtualization Empowered Metaverse: A Hierarchical Matching Approach," *in TechRxiv. Preprint,* https://doi.org/10.36227/techrxiv.24049998.v1, 2023, pp. 1-14, 2023.

[65] S. A. Kazmi et al., "Hierarchical matching game for service selection and resource purchasing in wireless network virtualization," *in IEEE Communications Letters,* vol. 22, no. 1, pp. 121–124, 2017.

[66] D. B. Rawat et al., "Fusion of software defined networking, edge computing, and blockchain technology for wireless network virtualization," *in IEEE Communications Magazine,* vol. 57, no. 10, pp. 50–55, 2019.

[67] Y. Li et al., "A load-balanced re-embedding scheme for wireless network virtualization," *in IEEE Transactions on Vehicular Technology,* vol. 70, no. 4, pp. 3761–3772, 2021.

[68] Y. Liu et al., "Slicing4Meta: An Intelligent Integration Architecture with Multi-Dimensional Network Resources for Metaverse-as-a-Service in Web 3.0," *in IEEE Communications Magazine,* vol. 61, no. 8, pp. 20-26, 2023.

[69] A. Mesodiakaki et al.,"ONE: Online Energy-efficient User Association, VNF Placement and Traffic Routing in 6G HetNets," *in IEEE Globecom Workshops (GC Wkshps),* Brazil, pp. 304-309, 2022.

[70] Y. Liu et al., "Virtual Network Embedding in Fiber-Wireless Access Networks for Resource-Efficient IoT Service Provisioning," *in IEEE Access,* vol. 7, pp. 65506-65517, 2019.

[71] C. D. Alwis et al., "A Survey on 6G Frontiers: Trends, Applications, Requirements, Technologies and Future Research," *in IEEE Open Journal of the Communications Society,* vol. 2, pp. 836-886, 2021.

[72] Hani Sami et al., "The Metaverse: Survey, Trends, Novel Pipeline Ecosystem and Future Directions," *IEEE Communications Surveys and Tutorials,* vol. xx, pp. xx-xx, 2024, doi: 10.1109/COMST.2024.3392642.

[73] D. Mourtzis et al., "Digital twin inception in the Era of industrial metaverse," *Front. Manuf. Technol.,* vol. 3, no. 1155735, 2023, pp. 1-8.

[74] B. Siniarski et al., "Need of 6G for the Metaverse Realization," *IEEE Access,* vol. 4, 2023, pp. 1-31.

[75] S. Zhang et al., "Towards Green Metaverse Networking: Technologies, Advancements and Future Directions," *in IEEE Network,* vol. 37, no. 5, pp. 223-232, 2023.

[76] Shah Zeb et al., "Industry 5.0 is Coming: A Survey on Intelligent NextG Wireless Networks as Technological Enablers," *in axXiv Preprint,* doi: 10.48550/arXiv.2205.09084, pp. 1-42, 2022.

[77] K. Li et al., "When Internet of Things Meets Metaverse: Convergence of Physical and Cyber Worlds," *in IEEE Internet of Things Journal,* vol. 10, no. 5, pp. 4148-4173, 2023.

[78] Y. Cai et al., "Compute- and Data-Intensive Networks: The Key to the Metaverse," *1st International Conference on 6G Networking (6GNet),* Paris, France, 2022, pp. 1-8.

[79] M. Ali et al., "Metaverse Communications, Networking, Security, and Applications: Research Issues, State-of-the-Art, and Future Directions," *in arXiv preprint,* arXiv:2212.13993, pp. 1-38, 2022.

[80] S. Karunarathna et al., "The Role of Network Slicing and Edge Computing in the Metaverse Realization," *in IEEE Access,* vol. 11, pp. 25502-25530, 2023.

[81] Y. Wang et al., "A Survey on Metaverse: Fundamentals, Security, and Privacy," *in arXiv preprint,* https://arxiv.org/abs/2203.02662, pp. 1-32, 2022.

[82] L. Chang et al., "6G-Enabled Edge AI for Metaverse: Challenges, Methods, and Future Research Directions," *Journal of Communications and Information Networks,* vol. 7, no. 2, pp. 1-15, 2022.

[83] L.U.Khan et al., "Machine Learning for Metaverse-enabled Wireless Systems: Vision, Requirements, and Challenges," *in arXiv preprint,* https://arxiv.org/abs/2211.03703, pp. 1-7, 2022.

[84] Huawei Huang et al., "Economic Systems in Metaverse: Basics, State of the Art, and Challenges," *in arXiv preprint,* https://arxiv.org/abs/2212.05803, pp. 1-19, 2023.

[85] Y. Hui et al., "Mobile Metaverse: A Road Map from Metaverse to Metavehicles," *in arXiV preprint,* arxiv-2309.09304, pp. 1-7, 2023.

[86] V. K. Quy et al., "Innovative Trends in the 6G Era: A Comprehensive Survey of Architecture, Applications, Technologies, and Challenges," *in IEEE Access,* vol. 11, pp. 39824-39844, 2023.

[87] F. Guo et al., "Enabling Massive IoT Toward 6G: A Comprehensive Survey," *in IEEE Internet of Things Journal,* vol. 8, no. 15, pp. 11891-11915, 2021.

[88] D. Zelenyanszki et al., "A privacy awareness framework for NFT avatars in the metaverse," *in International Conference on Computing, Networking and Communications (ICNC),* Honolulu, HI, USA, 2023, pp. 431-435.

[89] P. Bhattacharya et al., "Coalition of 6G and Blockchain in AR/VR Space: Challenges and Future Directions," *in IEEE Access,* vol. 9, pp. 168455-168484, 2021.

[90] Hao Yu et al., "Towards 6g-based metaverse: Supporting highly-dynamic deterministic multi-user extended reality services," *in IEEE Network,* vol. 37, no. 4, pp. 30-38, 2023.

[91] H. Moudoud et al., "Federated Learning Meets Blockchain to Secure the Metaverse," *in 2023 International Wireless Communications and Mobile Computing (IWCMC),* Marrakesh, Morocco, 2023, pp. 339-344.

[92] D. C. Nguyen et al., "6G Internet of Things: A Comprehensive Survey," *in IEEE Internet of Things Journal,* vol. 9, no. 1, pp. 359-383, 2022.

[93] A. H. Khan et al., "Blockchain and 6G: The Future of Secure and Ubiquitous Communication," *in IEEE Wireless Communications,* vol. 29, no. 1, pp. 194-201, 2022.

[94] I. Sarrigiannis et al., "Online VNF Lifecycle Management in an MEC-Enabled 5G IoT Architecture," *in IEEE Internet of Things Journal,* vol. 7, no. 5, pp. 4183-4194, 2020.

[95] Jing Yang et al., "Parallel Manufacturing for Industrial Metaverses: A New Paradigm in Smart Manufacturing," *IEEE/CAA Journal of Automatica Sinica,* Vol. 9, No. 12, pp. 2063-2070, 2022.

[96] Q. Qi et al., "Integration of Energy, Computation and Communication in 6G Cellular Internet of Things," *in IEEE Communications Letters,* vol. 24, no. 6, pp. 1333-1337, 2020.

[97] Yitong Wang et al., "A Survey of Mobile Edge Computing for the Metaverse: Architectures, Applications, and Challenges," *IEEE 8th International Conference on Collaboration and Internet Computing (CIC),* pp. 1-9, 2022.

[98] L. U. Khan et al., "Network Virtualization Empowered Metaverse: A Hierarchical Matching Approach," *TechRxiv Preprint,* https://doi.org/10.36227/techrxiv.24049998.v1, pp. 1-14, 2023.

[99] M. Xu et al., "A Full Dive Into Realizing the Edge-Enabled Metaverse: Visions, Enabling Technologies, and Challenges," *IEEE Communications Surveys and Tutorials,* vol. 25, no. 1, pp. 656-700, 2023.

[100] Wei Lim et al., "Realizing the Metaverse with Edge Intelligence: A Match Made in Heaven," *IEEE Wireless Communications,* vol. 30, no. 4, pp. 64-71, 2023.

[101] Chuan Chen et al., "Privacy Computing Meets Metaverse: Necessity, Taxonomy and Challenges," *Adhoc Networks,* vol. 158, pp. 1-14, 2024.

[102] Y. Wu et al., "Survey of Intelligent Network Slicing Management for Industrial IoT: Integrated Approaches for Smart Transportation, Smart Energy, and Smart Factory," *in IEEE Communications Surveys and Tutorials,* vol. 24, no. 2, pp. 1175-1211, 2022.

[103] Y. Liu et al., "Virtual Network Embedding in Fiber-Wireless Access Networks for Resource-Efficient IoT Service Provisioning," *in IEEE Access,* vol. 7, pp. 65506-65517, 2019.

[104] S. Yang et al., "Delay-Sensitive and Availability-Aware Virtual Network Function Scheduling for NFV," *in IEEE Transactions on Services Computing,* vol. 15, no. 1, pp. 188-201, 2022.

[105] S. K. Taskou et al., "Energy and Cost Efficient Resource Allocation for Blockchain-Enabled NFV," *in IEEE Transactions on Services Computing,* vol. 15, no. 4, pp. 2328-2341, 2022.

[106] Vincenzo Eramo et al., "Effectiveness of Segment Routing Technology in Reducing the Bandwidth and Cloud Resources Provisioning Times in Network Function Virtualization Architectures," *Future Internet,* MDPI, vol. 11, no. 71, pp. 1-20, 2019.

[107] S. Garg et al., "SDN-NFV-Aided Edge-Cloud Interplay for 5G-Envisioned Energy Internet Ecosystem," *in IEEE Network,* vol. 35, no. 1, pp. 356-364, 2021.

[108] G. Sun et al., "Energy-Efficient Provisioning for Service Function Chains to Support Delay-Sensitive Applications in Network Function Virtualization," *IEEE Internet of Things Journal,* vol. 7, no. 7, pp. 6116-6131,2020.

[109] A. S. D. Alfoudi et al., "An Efficient Resource Management Mechanism for Network Slicing in a LTE Network," *in IEEE Access,* vol. 7, pp. 89441-89457, 2019.

[110] A. Moubayed et al., "Softwarization, Virtualization, and Machine Learning for Intelligent and Effective Vehicle-to-Everything Communications," *in IEEE Intelligent Transportation Systems Magazine,* vol. 14, no. 2, pp. 156-173, 2022.

[111] H. Hantouti et al., "Service Function Chaining in 5G and Beyond Networks: Challenges and Open Research Issues," *in IEEE Network,* vol. 34, no. 4, pp. 320-327,2020.

[112] H. Hawilo et al., "Network Function Virtualization-Aware Orchestrator for Service Function Chaining Placement in the Cloud," *in IEEE Journal on Selected Areas in Communications,* vol. 37, no. 3, pp. 643-655, 2019.

[113] K. Kaur et al., "An Energy-driven Network Function Virtualization for Multi-domain Software Defined Networks," *IEEE INFOCOM Wkshps,* Paris, France, 2019, pp. 121-126, 2019.

[114] B. Nour et al., "Coexistence of ICN and IP Networks: An NFV as a Service Approach," *IEEE Global Communications Conference (GLOBECOM),* Waikoloa, HI, USA, 2019, pp. 1-6, 2019.

[115] G. Sun et al., "Low-Latency and Resource-Efficient Service Function Chaining Orchestration in Network Function Virtualization," *IEEE Internet of Things Journal,* vol. 7, no. 7, pp. 5760-5772, 2020.

[116] A. Mosayebi et al., "Heuristic Based Algorithm for SFC Allocation in 5G Experience Applications," *in 6th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS),* Mashhad, pp. 1-6, 2020.

[117] R. Souza et al., "NFV Data Centers: A Systematic Review," *in IEEE Access,* vol. 8, pp. 51713-51735, 2020.

[118] Esmaeil Amiri et al., "Deep Reinforcement Learning for Robust VNF Reconfigurations in O-RAN," *IEEE Transactions on Network and Service Management,* vol. 21, no. 1, pp. 1115-1128, 2024.

[119] Wen Wu et al., "AI-Native Network Slicing for 6G Networks," *IEEE Wireless Communications Magazine,* vol. 29, no. 1, pp. 96-103, 2022.

[120] M. Mosahebfard et al., "SDN/NFV-Based Network Resource Management for Converged Optical-Wireless Network Architectures," *21st International Conference on Transparent Optical Networks (ICTON),* Angers, France, pp. 1-4, 2019.

[121] R. B. Mulinteti et al., "An Efficient Delay Tolerance and Cost-Efective Secure NFV Enabled Multi-casting in Mobile Edge Cloud Networks," *Wireless Personal Communications,* vol. 126, 2022, pp. 1845-1862.

[122] J. Fu et al., "An Efficient VNF Deployment Scheme for Cloud Networks," *IEEE 11th International Conference on Communication Software and Networks (ICCSN),* Chongqing, China, 2019, pp. 497-502.

[123] H. Cao et al., "Resource-Ability Assisted Service Function Chain Embedding and Scheduling for 6G Networks With Virtualization," *in IEEE Transactions on Vehicular Technology,* vol. 70, no. 4, pp. 3846-3859, 2021.

[124] L. Dong et al., "On Application-Aware and On-Demand Service Composition in Heterogenous NFV Environments," *IEEE Global Communications Conference (GLOBECOM),* Waikoloa, HI, USA, pp. 1-6, 2019.

[125] Y. Anser et al., "Energy-Aware Service Level Agreements in 5G NFV architecture," *8th International Conference on Future Internet of Things and Cloud (FiCloud),* Rome, Italy, 2021, pp. 377-382.

[126] M. Huang et al., "Reliability-Aware Virtualized Network Function Services Provisioning in Mobile Edge Computing," *in IEEE Transactions on Mobile Computing,* vol. 19, no. 11, pp. 2699-2713, 2020.

[127] M. Zoure et al.,"Network Services Anomalies in NFV: Survey, Taxonomy, and Verification Methods," *IEEE Transactions on Network and Service Management,* vol. 19, no. 2, pp. 1567-1584, 2022.

[128] A. Mesodiakaki et al., "ONE: Online Energy-efficient User Association, VNF Placement and Traffic Routing in 6G HetNets," *IEEE Globecom Workshops (GC Wkshps),* Rio de Janeiro, Brazil, pp. 304-309, 2022.

[129] L. Gao et al., "Service Chain Rerouting for NFV Load Balancing," *IEEE Global Communications Conference,* Taipei, Taiwan, 2020, pp. 1-6.

[130] G. Wang et al., "Service Function Chain Planning with Resource Balancing in Space-Air-Ground Integrated Networks," *IEEE Global Communications Conference (GLOBECOM),* Waikoloa, HI, USA, 2019, pp. 1-6.

[131] G. Wang et al., "SFC-Based Service Provisioning for Reconfigurable Space-Air-Ground Integrated Networks," *IEEE Journal on Selected Areas in Communications,* vol. 38, no. 7, pp. 1478-1489, 2020.

[132] Amreen et al., "Performance Evaluation in Cloud Computing Model using Queuing Models," *International Journal of Grid and Distributed Computing,* vol. 10, no. 3, pp. 15–24,2017.

[133] U. Drolia et al., "Cachier: edge-caching for recognition applications," *in Proc. IEEE ICDCS*, arxiv preprint, pp.276–286, 2021.

[134] Q. Xu et al., "Blockchain-Based Trustworthy Edge Caching Scheme for Mobile Cyber-Physical System," *in IEEE IoT Journal,* vol. 7, no. 2, no. xx, pp. 1098-1110, Feb. 2020.

[135] W. Zhang et al., "Collaborative Task Execution in Mobile Cloud Computing Under a Stochastic Wireless Channel," *IEEE Transactions on Wireless Communications*, vol. 14, no. 1, pp. 81–93, 2015.

[136] H. Zhang et al., "Toward Vehicle-Assisted Cloud Computing for Smartphones," *IEEE Transactions on Vehicular Technology*, Vol. 64, No. 12, pp. 5610-5618, 2015.

[137] Y. Yang et al., "Joint Optimization of Energy Consumption and Packet Scheduling for Mobile Edge Computing in Cyber-Physical Networks," *in IEEE Access,* vol. 6, pp. 15576-15586, 2018.

[138] H. Mughal et al., "Efficient Allocation of Resource-Intensive Mobile Cyber–Physical Social System Applications on a Heterogeneous Mobile Ad Hoc Cloud," *in IEEE Transactions on Network Science and Engineering,* vol. 9, no. 3, pp. 958-969, 2022.

[139] Xinyu Zhou et al., "Mobile Augmented Reality with Federated Learning in the Metaverse," *in arXiv preprint,* pp. 1-7, 2022, arXiv:2212.08324.

[140] C. T. Nguyen et al., "Metachain: A novel blockchain-based framework for metaverse applications," *in IEEE 95 th vehicular technology conference,* pp. 1-5, 2022, arXiv preprint arXiv:2201.00759.

[141] Y. Gui et al., "A Cache Placement Strategy Based on Compound Popularity in Named Data Networking," *in IEEE Access*, vol. 8, pp. 196002-196012, 2020.

[142] E. Silva et al., "NDN Content Store and Caching Policies: Performance Evaluation," *in Computers, MDPI*, vol. 11, no. 37, pp. 1-17, 2022.

[143] Jintao Liu et al., "Multi-objective Model and Genetic Algorithm for Multisource Multicast VNF Service Chain Deployment Problem," *IAENG International Journal of Computer Science*, vol. 51, no. 6, pp. 562-571, 2024.

[144] Hejun Xuan et al., "Novel Virtual Network Function Service Chain Deployment Algorithm based on Q-learning," *IAENG International Journal of Computer Science*, vol. 50, no.2, pp. 736-744, 2023.

Mahfuzulhoq chowdhury received his Ph.D. Degree (doctoral) in Telecommunications at the University of Quebec, INRS, Montreal, Canada in 2019. He received his master's and bachelor's degrees in Computer Science and Engineering (CSE) in 2010 and 2015, respectively. He is a faculty member at the Department of Computer Science and Engineering, Chittagong University of Engineering and Technology since September 2010. His research interests are related to resource allocation in wireless networks, Software-defined networking (SDN), Network function virtualization (NFV), Space-air-ground integrated networks (SAGIN), digital twin, optimization, cloud computing, future generation networks, machine learning, deep learning, artificial intelligence, blockchain, network security, IoT, UAV, VANET, industry 5.0, metaverse, XR technology, 6G, blockchain, immersive telepresence, extended reality, mobile application, and high-speed communication, among others. He has published several research papers in different IEEE journals, transactions, magazines, and conferences as well as chapters of books.