This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

When Computing Follows Vehicles: Decentralized Mobility-Aware Resource Allocation for Edge-to-Cloud Continuum

Zeinab Nezami, Emmanouil Chaniotakis, and Evangelos Pournaras

***Abstract*—The transformation of smart mobility is unprecedented–Autonomous, shared and electric connected vehicles, along with the urgent need to meet ambitious net-zero targets by shifting to low-carbon transport modalities result in new traffic patterns and requirements for real-time computation at large-scale, for instance, self-driving cars and augmented reality applications. The cloud computing paradigm can neither respond to such low-latency requirements nor adapt resource allocation to such dynamic spatio-temporal service requests. This paper addresses this grand challenge by introducing a novel decentralized optimization framework for mobility-aware edge-to-cloud resource allocation, service offloading, provisioning and load-balancing. In contrast to related work, this framework comes with superior efficiency and cost-effectiveness under evaluation in real-world traffic settings and mobility datasets. This breakthrough capability of '*computing follows vehicles*' proves to be superior to balance resource utilization in a highly cost-effective way, while preventing service deadline violations by 14%-34%.**

***Index Terms*—Edge-to-Cloud Computing, Smart Mobility, Distributed Optimization, Dynamic Resource Allocation, Multi-agent System.**

I. INTRODUCTION

IN 2019, the UK domestic transport sector emerged as the largest source of greenhouse gase emissions, contributing to 27% of the country's total emissions[1], which emphasizes the urgent need for solutions that reduce CO2 emission, particularly through smart traffic management and mobility services. As Intelligent Transportation Systems (ITS) advance, technologies such as connected and autonomous vehicles are envisioned to provide a safer and environmentally friendly transportation ecosystems [1], [2]. These vehicles are increasingly integrated with wireless communication capabilities, enabling a plethora of applications from rerouting and road safety to location-based services [3], [4]. Furthermore, the imminent proliferation of Internet of Things (IoT) devices, including vehicles, is anticipated to surpass 55.7 billion by 2025, generating up to 79.4 zettabytes of data [5]. This surge in data generation presents a dual challenge: it coincides with a dramatic increase in the energy consumption of Information and Communication Technology (ICT) systems, which currently account for over 10% of global energy consumption and are projected to exceed 20% by 2030 [6]. These trends reveal a critical issue: how can ICT infrastructure efficiently manage this immense data growth and energy demand, particularly within the context of smart mobility? This paper addresses this emerging challenge by focusing on the intersection of ICT and mobility, an area that has yet to receive adequate attention in the literature.

The surge in data generation is further compounded by the vehicle mobility, a defining characteristic of the IoT environment, which introduces significant dynamism and uncertainty. This dynamic nature presents challenges for managing ICT resources, especially for applications that demand strict Quality of Service (QoS) guarantees. Existing Vehicle-to-Cloud (V2C) and Vehicle-to-Vehicle (V2V) architectures often struggle to fulfill the diverse requirements of mobility services, particularly in terms of latency, efficiency, and scalability [7]. To address these limitations, Vehicle-to-Infrastructure (V2I) systems improve data distribution by bringing cloud services closer to the network edges. Fog and edge computing [8]–[10] offer a novel approach by offloading computation from vehicles to local servers, enabling more efficient data sharing [11] and traffic rerouting [12], [13]. The distributed edge-to-cloud architecture reduces data travel across the network, resulting in energy savings ranging from 14% to over 80% compared to fully centralized cloud models [14], [15].

This paper introduces a novel service provisioning framework designed to address the complexities of the evolving mobility ecosystem. The distributed and heterogeneous nature of edge-to-cloud infrastructure poses challenges, particularly in hosting and managing services under the dynamic and stochastic conditions of vehicular networks. The placement of smart mobility services near intersections or vehicle destinations, intended to meet strict QoS requirements, can lead to server overloading and inefficient resource use, with some services becoming over-provisioned, while others remain underserviced. Beyond balancing the workload and meeting QoS needs, multiple optimization factors must be considered to efficiently solve the service provisioning problem, especially in light of the mobility patterns and traffic dynamics of vehicles [16], [17]. Current research on service provisioning within edge-to-cloud networks has explored optimization techniques in static and dynamic environments, emphasizing criteria such as QoS, energy consumption, and resource efficiency, while

Zeinab Nezami is a Postdoctoral Researcher at the School of Electronic and Electrical Engineering, University of Leeds, Leeds, United Kingdom (e-mail: z.nezami@leeds.ac.uk). Emmanouil Chaniotakis is an Associate Professor with the UCL Energy Institute, University College London, London, United Kingdom (e-mail: m.chaniotakis@ucl.ac.uk). Evangelos Pournaras is a Professor at the School of Computer Science, University of Leeds, Leeds, United Kingdom (e-mail: e.pournaras@leeds.ac.uk).

[1]Transport energy and environment statistics, available at https://www.gov.uk/government/statistics/transport-and-environment-statistics-2021 (accessed April 2024).

recognizing the need for holistic approaches that address the complexities of mobility, provisioning cost, and dynamic resource availability [18], [19].

This paper introduces a novel distributed service provisioning framework that dynamically balances the competing demands of QoS, service provisioning cost, and sustainability within the edge-to-cloud ecosystem. The proposed framework features a reconfigurable infrastructure that enables real-time activation/deactivation of servers based on service demand, thus efficiently managing smart mobility services, while responding to dynamic spatio-temporal mobility patterns. The main contributions of this work include: (i) the development of a novel open-source framework[1] that enables dynamic deployment and migration of mobility services, (ii) the formulation of an optimization problem that addresses both QoS and mobility-aware service placement through three system-wide objectives and eight local objectives, (iii) the application of a collective decision-making algorithm to solve this problem effectively, (iv) the integration of real-world data, simulation and emulation environments for ICT and mobility that expand the realism of experimentation found in earlier work, (v) the creation of a comprehensive open-source benchmarking dataset that encompasses the Munich edge-to-cloud infrastructure and IoT workloads[2], facilitating further research in this domain, and (vi) extensive evaluations using the real-world data to demonstrate the cost-effectiveness and scalability of the proposed solution.

This paper is organized as follows: The next section highlights our contribution to the existing literature. The proposed framework is introduced in Section III, followed by the formulation of the service placement problem in Section IV. Then, a cooperative multi-agent approach for dynamic service placement is presented in Section V. Finally, Sections VI and VII provide experimental evaluations and conclusions, along with suggestions for future research.

II. Related Work

Service provisioning within edge-to-cloud network has garnered significant attention, leading to extensive exploration in both static and dynamic environments. Various optimization methods have been employed, including single [20], [21] and multi-objective [22], [23] optimization, linear programming [24], [25], Markov Decision Process (MDP) [23], [26], game theory [27], [28], and fuzzy logic [29], catering to diverse QoS criteria such as service delay [25], [28], [30]–[33], resource efficiency [31], energy consumption [32]–[36], economic cost [25], [32], [37], and system utility [20], [38]. These formulations lay the groundwork for developing service placement algorithms falling into categories such as exact [39], [40], heuristic [21], [41]–[43], meta-heuristic [30], [33], [36], and machine learning-based [19], [37], [41], [44], [45] solutions. Static placement [24], [26], [29], [31], [36], [39], [40] becomes impractical over time in dynamic environments due to the impact of mobility on response time and infrastructure energy consumption. Especially in mobile scenarios that require stateful handling, migration becomes essential for optimizing efficiency. This section summarizes recent literature addressing the problem within a mobility-aware context, as detailed in Table I. Zhan *et al.* [47] introduce an MDP-based model to address service placement challenges in Vehicular Edge Computing (VEC) scenarios, focusing on reducing processing delay and system energy consumption. The authors leverage Deep Reinforcement Learning (DRL) methods to optimize this model. Awada *et al.* [46] propose an integrated Mobile Edge Computing (MEC) framework for offloading computation-intensive applications to edge devices. This framework consolidates edge resources from diverse locations into a unified pool, facilitating comprehensive monitoring from a single control plane. Fan *et al.* [42] present a joint service offloading and resource allocation scheme for vehicles assisted MEC scenarios, aiming to minimize total processing delay through a two-stage heuristic algorithm that offers near-optimal solutions with low computational complexity. Ouyang *et al.* [28] introduce a centralized MDP approximation method and a distributed non-cooperative approach based on game theory for dynamic service placement to optimize communication and computing delay in mobile edge environments. The authors assume that wireless connections between users and edge nodes remain unchanged between placement rounds. Liu *et al.* [37] propose a vehicle-edge-cloud resource allocation framework aimed at maximizing system utility. This framework supports V2V offloading to enable vehicles with idle resources to engage in service deployment, and V2I offloading for load balancing. They employ a multi-agent MDP-based DRL algorithm to make optimal scheduling decisions.

The current research [23], [28], [33], [37], [41], [42], [44], [46]–[48] on service placement typically focuses on a single or limited criteria. Service placement is inherently a trade-off problem, necessitating a comprehensive examination of multiple, often conflicting objectives. This research studies a holistic perspective encompassing sustainability, QoS, service cost, and resource efficiency within the heterogeneous edge-to-cloud infrastructure. Application placement in a fog network is dynamic and autonomous, unlike the cloud, as fog nodes can join and leave the network freely. This variability in resource availability is compounded by fluctuations in demand for resources based on incoming request volume. Adjusting application placement at runtime to accommodate these changes is crucial, an aspect overlooked in current literature but addressed in this paper. This work supports adaptation to the V2I resources availability and service demands, ensuring flexibility and responsiveness to dynamic network conditions. Additionally, previous studies [33], [37], [42], [46] predominately rely on central control, planning and decision making based on non-cooperative solutions [28], our approach stands out for its distributed cooperative nature, offering scalability, resilience, privacy preservation and flexibility. This research leverages the collective learning approach of EPOS (Economic Planning and Optimized Selections) [49], [50], a state-of-the-art decentralized multi-agent system designed to tackle complex multi-objective discrete-choice combinatorial problems, thereby supporting cooperative optimization through a

[1]MERA framework, available at https://github.com/DISC-Systems-Lab/Edge-Mobility-Cooptimization

[2]Munich IoT Benchmarking Dataset, available at https://github.com/Znbne/MunichIoT

TABLE I: Literature review summary

Criteria/ Study	Optimization Criteria	Proposed Approach	IoT Load	ICT Network	Traffic Network (mobility)
[37]	SU	DRL	Synthetic	Synthetic	-
[46]	SD, RU	Bin-Packing	Alibaba cluster trace	-	Synthetic
[42]	SD	Heuristic	Synthetic	-	Synthetic
[44]	SD	DRL	-	Synthetic	Rome (GPS trajectory)
[33]	EC, SD	Greedy, Genetic algorithm	Synthetic	-	San Francisco (real trajectory)
[47]	SD, EC	DRL	Synthetic	-	-
[48]	SR	Approximation	Synthetic	Synthetic	Melbourne (real trajectory)
[28]	CD, MC	Approximation, Greedy	Synthetic	Synthetic	Helsinki (shortest paths)
This work	EC, SD, RU, XC, CF	Collective learning	MAWI IoT trace	Munich edge-core-cloud network	Munich (shortest/default trajectory)

Abbreviation: SU - System Utility, RU - Resource Utilization, SD - Service Delay, EC - Energy Consumption, XC - Execution Cost, CF - CO2 Footprint, SR - System Robustness, MC - Migration Cost.

structured agents interaction model.

III. Service Provisioning Framework

This work defines a novel challenge in the realm of computing-mobility and tailors the system proposed in our prior work [31], offering a comprehensive solution finely tuned to mobility dynamics. We present a Mobility-aware Edge-to-cloud Resource Allocation framework, MERA, that dynamically incorporates resource availability, mobility, and QoS considerations to address the challenges of service provisioning. The framework assesses resource utilization across the edge-to-cloud continuum by balancing factors such as load distribution, QoS fulfillment, economic cost, energy efficiency, and sustainability to meet the requirements of smart mobility services, particularly those requiring low-latency and high-bandwidth capabilities. Improving QoS is particularly advantageous for applications such as real-time data analytics, augmented/virtual reality, industrial control with ultra-low latency, and big data streaming [24]. This section outlines the motivational applications of MERA that are driving advancements in automotive and telecommunications technologies, along with a discussion of its layered architecture. Following this, the next section delves into problem formulation, considering both the individual and system-wide objectives of end-users, service providers, and ICT infrastructure providers.

A. Motivational Application

Based on the latest Cellular-Vehicle-to-Everything (C-V2X) roadmap of 5GAA[1], the automotive and telecommunications industries are racing to enable technologies and applications such as HD Maps for fully driverless cars. These maps (e.g., Civil Maps[2]) provide enhanced precision and accuracy, dynamically updated to reflect near real-time environmental conditions. Major players such as Google, Ford, BMW, Tesla, and General Motors are investing in HD Maps to advance autonomous vehicles. HD Maps aims to create a system where vehicles benefit from aggregated data on traffic patterns and construction zones, gathered by vehicles and sent over cellular networks. This enables commuters to contribute real-time incident reports to a precise crowd-sourced map accessible to all. Vehicles utilize these data via 4G LTE connectivity for automated mapping, overcoming challenges posed by frequent infrastructure changes, which traditional mapping struggles to address due to cost limitations in wireless data upload and processing.

In line with the development of HD Maps, ABI Research[3] forecasts a significant rise in Augmented Reality Head-Up Displays, with an estimated 15 million units shipped by 2025, including 11 million integrated into vehicles. This paper focuses on passenger-facing HD Maps of augmented reality technology, resembling the 2020 Mercedes-Benz GLE display [51]. Continuous data flows from vehicles to servers hosting AR services incur significant communication and processing costs, demanding substantial computational resources and low-latency processing [52]. This poses challenges to the readiness of self-driving cars. We assume that all considered vehicles are equipped with onboard cameras, continuously uploading captured video/images to the edge-to-cloud infrastructure. Edge-to-cloud servers analyze data streams, providing updated maps to clients. The Sense-Process-Actuate model serves as the foundational framework for this application, starting with sensor data collection as a continuous stream to higher-layer computing nodes for processing, and ending with command transmission to actuators [53], [54].

B. Distributed Architecture

The system architecture of MERA within the V2I communication model is illustrated in Figure 1. The bottom layer encompasses the Traffic Network, serving as the infrastructure for delivering smart mobility services to end-users. This layer includes both congested and uncongested road networks, along with the mobility profiles of vehicles and supporting tools. Above the Traffic Network, the Data Layer serves as a foundation that enhances the framework realism by incorporating for the first time various datasets, tools, and emulators. This layer is essential for emulating real-world communication settings pertinent to LTE access/edge networks, core networks, cloud data centers, and IoT workloads. It focuses on providing the necessary data to accurately reflect IoT traffic dynamics and network interactions. Such realism has been so far a major limitation in earlier work.

Positioned above the Data Layer, the ICT Network represents a structured architecture that defines how smart mobility services are hosted and managed. It integrates the access network, core network, and cloud data centers, leveraging the

[1] 5G Automotive Association, available at https://5gaa.org/about-us/ (accessed April 2024).

[2] Civil Maps, available at https://civilmaps.com/ (accessed April 2024).

[3] ABI research, available at https://www.abiresearch.com/press/augmented-reality-redefine-automotive-user-interfa/ (accessed April 2024).

data from the Data Layer to establish a cohesive network infrastructure. This layer utilizes networking and computation nodes across the edge-to-cloud hierarchy, ensuring seamless operation among vehicles, fog servers, and cloud centers. Its primary function is to facilitate efficient data processing and communication, built upon the realistic scenarios established by the Data Layer. In the context of fog computing literature [54], [55], fog nodes refer to computing or networking resources positioned between a data source and the central cloud. These nodes, which can include smartphones and routers, may be deployed at the cellular base station sites or data aggregation points such as a routers at the edge of the core network. The connectivity infrastructure is established by cellular LTE base stations (referred to as access points), edge and core routers, and switches. Access points establish connections to the Internet via edge/core routers. Fog servers are interconnected with access points through edge routers, while the cloud is accessed through core routers [56]. Vehicles are equipped with a cellular radio interface [57] to establish a connection and join the network upon entering the coverage range of a fixed access point. Access points, equipped with 4G LTE modules, connect to core/cloud nodes via a Wide Area Network, and they are interlinked through wireless backhaul.

At the apex of the architecture sits the service placement strategy, which orchestrates edge-to-cloud resources based on traffic patterns and service demands. This strategy plays a crucial role in optimizing edge-to-cloud resource utilization and ensuring the efficient delivery of smart mobility services within the edge-to-cloud infrastructure.

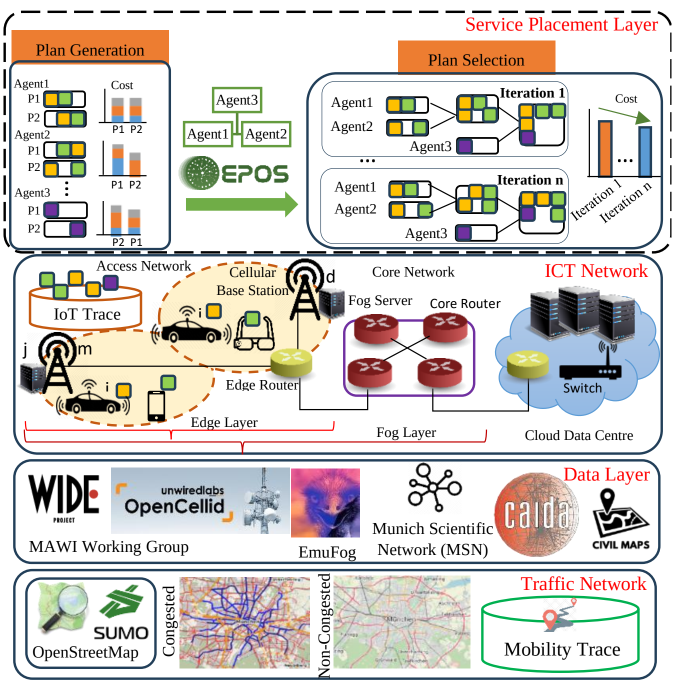

Fig. 1: Layered core components of MERA framework, consisting of: (1) Real-world traffic network, (2) Real-world data layer, (3) Vehicle-fog-cloud connectivity and computation infrastructure, and (4) Cooperative service placement orchestrating resource allocation.

C. Handling Requests and practicality

Any vehicle requesting smart mobility services sends a service request, containing parameters outlined in Table VI, along with its basic information (e.g., camera image, IP, mobility profile including velocity, timestamp, the orientation of the vehicle, and GPS coordinates) to its communicating fog server (which is the server connected to the connecting access point of the vehicle) for processing. Each fog server is equipped with a software agent responsible for decision making regarding the placement of the requests received by the server itself. The placement decision may be determined by considering the information about the agent's neighborhood and exchanged information with neighboring servers. This process ensures that provisioning objectives are met while adhering to a service level agreement (SLA). If the decision favors local execution, the request is processed by the node; otherwise, the task is offloaded to a selected fog/cloud server, where the choice is made based on real-time resource availability and network conditions. Upon acceptance, a container is instantiated at the designated server for request processing [58]. The mapping of services to the available servers (i.e., service placement) is updated periodically as vehicles move around. Upon completion of service computation, host servers transmit the result to the nearest access point, which relays the response back to the vehicles.

IV. Mobility-aware Service Placement Problem Formulation

This section formally defines the mobility-aware service provisioning problem as an online computational problem to solve modeled byquadratic cost functions, acknowledging its NP-hard nature [31], [59]. We formulate the problem using a set of (i) *system-wide objectives*, emphasizing workload balance, incentivization of renewable energy sources, and infrastructure energy efficiency, as well as (ii) *local objectives*, focusing on minimizing service provisioning costs and meeting QoS requirements. The solution to the service provisioning problem is a service placement plan (δ) that contains placement decisions (i.e., binary variables), which place each service either on a fog server or on a cloud center. The binary variables x_{ij} and x_{ic} denote whether service i is placed on the fog node j or the cloud node c, respectively. $\overline{X}_{ij}$ denotes the initial configuration of i on j, which indicates whether j currently hosts the service. We consider a discrete time-slotted system model, where time is divided into slots, and services are generated at the beginning of each slot. Then each time slot becomes a decision round. Hence, we denote both time slot and decision round as τ in seconds. Table VI lists the notations used in the problem formulation.

The proposed framework is highly practical, operating without assumptions and requiring only minimal information about mobile nodes. For service placement, only the incoming resource demand from mobile nodes to fog nodes is required. Additionally, the framework may require the average transmission rate and propagation delay, approximated by the round-trip delay measured through a ping mechanism, between end-devices and their corresponding access point. The system has

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

the flexibility to either own its fog resources or acquire them through rental arrangements with edge network providers such as AT&T, Nokia, Verizon, or other edge resource owners [60]–[62]. This work comes for the first time with high modeling rigor, grounded in real-world traffic and ICT architecture settings, specifically tailored for the vehicle-to-edge-to-cloud network paradigm. The quality and realism of the data utilized in this study is ensured through selection from reputable sources, as detailed in Section VI.

A. System-wide Objectives

Due to the dynamic nature of vehicle mobility and the inherent unpredictability of the wireless medium, mobile IoT networks experience fluctuations in traffic load [63]. This research aims to assist policy makers and service providers in managing energy consumption and available edge-to-cloud capacity to handle incoming traffic effectively. Switching nodes off/on is a prevalent strategy for network management and resource optimization, providing various advantages such as improved energy efficiency, reduced operational costs, enhanced fault tolerance, and support for network scalability. However, this approach presents challenges in dynamic application placement, load balancing, and managing temporary network disruptions, especially in the time-varying environment of transport networks. Careful planning and monitoring are crucial for successful implementation. This research proposes a load-balancing strategy involving selectively deactivating network nodes and redistributing the workload among the remaining active nodes. Furthermore, this research objective extends beyond merely achieving workload balance and reducing energy consumption; it also aspires to advance renewable energy utilization through incentivization strategies. Energy harvesting from renewable resources such as wind and solar power holds significant promise for enhancing the efficiency and longevity of edge-to-cloud networks while also fostering sustainability from an environmental perspective [64].

1) Workload Balance: Leveraging fog nodes in transport environments enhances resource efficiency within edge-to-cloud networks and facilitates the execution of time-sensitive services. However, the individual objectives of fog agents may oppose both the broader system-wide goals and the interests of fog service providers or vehicles. Opting for the placement plans with the lowest costs can lead to the tragedy of the commons, where self-interested agents favoring the nearest edge servers may encounter elevated execution costs due to an uneven distribution of edge-to-cloud resource utilization. Effective load balancing, which mitigates the risk of bottlenecks, ensures a more robust and adaptable network infrastructure. By evenly distributing workload across the network, fog servers remain highly available to efficiently handle incoming requests, thereby reducing the likelihood of delayed responses to critical situations [65]–[67]. The overarching objective is to minimize variance in server utilization (MIN-VAR), ensuring equitable load distribution across the network. Network nodes possess diverse capacities, necessitating the consideration of workload-to-capacity ratios to quantify node utilization. Equation (1) illustrates the degree of workload balance across all nodes, measured by the variance in terms of CPU and RAM usage:

$$G = Min\sqrt{\frac{\sum_{j=1}^{|C\cup F|}\left(\left(\frac{\zeta_j^p}{F_j^{\mathsf{p}}}-\frac{\overline{\zeta_j^p}}{F_j^{\mathsf{p}}}\right)^2+\left(\frac{\zeta_j^m}{F_j^{\mathsf{m}}}-\frac{\overline{\zeta_j^m}}{F_j^{\mathsf{m}}}\right)^2\right)}{2\cdot(|F|+|C|)}}, \tag{1}$$

where ζ_j^m and ζ_j^p denote the RAM and CPU load on node j and measured as $\sum_{i=1}^{|A|} L_i^{\mathsf{m}}\cdot x_{ij}$ and $\sum_{i=1}^{|A|} L_i^{\mathsf{p}}\cdot x_{ij}$, respectively.

2) Energy Consumption versus Workload Balance: The framework enables dynamic activation and deactivation of fog servers based on service demand, while also allowing for the adjustment of node participation levels in service hosting. To achieve this, an *Infrastructure Planner* is defined, integrating traffic monitoring agents to observe incoming traffic rates [68]. These agents, typically associated with Software Defined Networking (SDN) controllers such as OpenDayLight and ONOS, monitor aggregated IoT request traffic rates to fog nodes via northbound APIs [69], [70]. Using incoming traffic data and QoS considerations, the planner decides on activating or deactivating machines within the network, utilizing a resource configuration vector as a reference for fog servers. In response to increased demand, it may activate new machines in corresponding areas, while in regions with lower demand, it strategically powers down machines to conserve resources and minimize energy consumption. Consequently, the fog agents utilize the reference vector for conducting service placement and distributing the load on the active nodes.

3) Renewable Energy Incentives: Renewable energy can fully support the reliable operation of edge devices using a microgrid (solar-wind hybrid energy system) [71]. This highlights the potential of renewable energy sources to sustain edge device functionality. It is assumed that edge-to-cloud servers and their associated networking devices are partially powered by clean energy, utilizing energy storage devices [72] to maximize renewable energy utilization. Nodes with a higher proportion of clean energy are proposed to be given priority in service deployment. To this end, Equation (2) minimizes the squared Euclidean distance, Root Mean Square Error (RMSE), between network nodes' utilization and a target incentive vector based on the ratio of energy supplied from renewable sources.

$$G = \min\sqrt{\frac{\sum_{j=1}^{|F\cup C|}\left(\left(\frac{\zeta_j^p}{F_j^{\mathsf{p}}}-p_j^{\mathsf{r}}\right)^2+\left(\frac{\zeta_j^m}{F_j^{\mathsf{m}}}-p_j^{\mathsf{r}}\right)^2\right)}{2\cdot(|C|+|F|)}} \tag{2}$$

B. Local Objective

The deployment of an application across the edge-to-cloud network can significantly influence its non-functional aspects such as operational costs and performance [73]. The overall cost of executing a service placement plan δ in edge-to-cloud infrastructure encompasses the following components: processing cost (O_δ^{P}), RAM usage cost (O_δ^{M}), storage usage cost (O_δ^{S}), container deployment cost (O_δ^{D}), communication cost (O_δ^{C}), energy consumption cost (O_δ^{E}), and carbon footprint cost (O_δ^{F}). This broad modeling spectrum is one of the novelties of this work. As formulated in Equation (3), this work combines these parameters with a penalty cost for service delay (O_δ^{V}), serving as a QoS measurement. All the eight local cost functions are considered; however, depending on the scenario, some costs may be prioritized or not. This flexibility is achieved by adjusting the weight of each cost component using a multiplicative factor (e.g., γ_{P}, γ_{M}), allowing the system to prioritize certain objectives when necessary. Other costs can also be identified such as security safeguards which are not the focus of this paper.

$$\begin{aligned} L_\delta =& \gamma_{\mathsf{P}} O_\delta^{\mathsf{P}} + \gamma_{\mathsf{M}} O_\delta^{\mathsf{M}} + \gamma_{\mathsf{S}} O_\delta^{\mathsf{S}} + \gamma_{\mathsf{D}} O_\delta^{\mathsf{D}} + \gamma_{\mathsf{C}} O_\delta^{\mathsf{C}} + \\ & \gamma_{\mathsf{E}} O_\delta^{\mathsf{E}} + \gamma_{\mathsf{F}} O_\delta^{\mathsf{F}} + \gamma_{\mathsf{V}} O_\delta^{\mathsf{V}}, \end{aligned} \tag{3}$$

where $\sum_i \gamma_{\mathsf{i}} = 1$, $0 \leq \gamma_{\mathsf{i}} \leq 1$, and each factor γ_{P}, γ_{M}, ..., γ_{V} allows for fine-tuning the priority of each cost in the cost function, enabling scenario-specific customization of the objective.

1) Cost of Processing and Memory Resources: The processing cost in each node is variable and can differ from that of other nodes. Equation (4) quantifies the processing cost associated with the placement plan δ.

$$O_\delta^{\mathsf{P}} = \sum_{i=1}^{|A|} \sum_{j=1}^{|F \bigcup C|} x_{ij} \cdot z_{ij} \cdot L_i^{\mathsf{p}} \cdot C_j^{\mathsf{p}} \cdot \tau \tag{4}$$

Similar to the processing cost, the memory cost for each node is specific to that node. Equations (5) and (6) calculate the RAM and storage cost for the plan δ respectively.

$$O_\delta^{\mathsf{S}} = \sum_{i=1}^{|A|} \sum_{j=1}^{|F \bigcup C|} x_{ij} \cdot (L_i^{\mathsf{s}} \cdot C_j^{\mathsf{s}}) \cdot \tau \tag{5}$$

$$O_\delta^{\mathsf{M}} = \sum_{i=1}^{|A|} \sum_{j=1}^{|F \bigcup C|} x_{ij} \cdot (L_i^{\mathsf{m}} \cdot C_j^{\mathsf{m}}) \cdot \tau \tag{6}$$

2) Cost of Service Delay: IoT applications such as augmented reality are latency sensitive and have very rigid latency constraints in the order of tens of milliseconds [74], so that a low latency is crucial to ensure an acceptable quality of experience. The service delay for an IoT service is defined as the time span between the moment an end-device sends its request and the moment it receives the response for that request. The binary variable v_{ij} indicates whether the service delay e_{ij} for service i on server j violates the SLA-defined delay threshold h_i.

$$v_{ij} = \begin{cases} 0 & \text{if } e_{ij} < h_i \\ 1 & otherwise \end{cases} \tag{7}$$

The delay of the service i includes three components as *propagation delay*, *transmission delay*, and *waiting time* (processing delay plus queuing delay) [24], [75]. Since vehicles may have different data generation rates, the time taken to execute a service depends on the data rate and the available computing capacity of the server to which it is assigned. In contrast to the computation time, which remains relatively constant, communication time varies over time due to fluctuations in latency along the communication path between mobile devices and the infrastructure. As shown in Figure. 1, the mobility of vehicle i may result in dynamic changes in the communication links between the device and its service host node j over time. These changes occur due to connection handoffs and handovers [76]. Consequently, the dynamic nature of connections results in varying propagation delays and transmission rates for services within a given time interval. To address this challenge in measuring service delay, we calculate those two parameters by considering the connecting access points along the path of the vehicle during a specific time interval. The delay for a particular service is then computed using estimated values for propagation delay and transmission rate as follows:

$$\begin{aligned} e_{ij} =& w_{ij} + \sum_{m=1}^{|M|} P_{im} \cdot (2 \cdot (d_{im} + d_{mj}) \\ & + (\frac{q_i}{b_{im}^{\mathsf{u}}} + \frac{a_i}{b_{im}^{\mathsf{d}}}) + (\frac{q_i}{b_{mj}^{\mathsf{u}}} + \frac{a_i}{b_{mj}^{\mathsf{d}}})), \end{aligned} \tag{8}$$

where P_{im} signifies the probability of vehicle i being within the coverage zone of access point m during the time interval τ in a 2D area [77]. It is calculated as follows:

$$P_{im} = \frac{1}{\tau} \cdot (\frac{l_{im}}{s_i} - t_{im} - t_{im}^{\mathsf{w}}), \tag{9}$$

where l_{im} represents the coverage area (Euclidean distance) between device i and access point m on street x where i is moving, s_i denotes the speed of i, t_{im} indicates the timestamp of the initial connection (registration time) of i with m before sending the service request, and t_{im}^{w} denotes the waiting time for i to receive a reply after sending the service request to access point m.

To measure the waiting time (w_{ij}) of service i hosted on node j we adopt a multi-server M/M/C queuing model (i.e., Erlang-C) [78] for fog servers. Fog node j has n_j processing units, each with service rate μ_j (total processing capacity or service rate of node j is $F_j^p = n_j \cdot \mu_j$). For simplicity but without loss of generality, we assume that a cloud server similarly comprises of a set of homogeneous computing machines with similar configurations, such as CPU frequency. As a result, the computation delay for service i hosted on either a fog or cloud server j is measured using Equation (10) [24], [79].

$$w_{ij} = \frac{\mathrm{P}_{ij}^{\mathsf{Q}}}{F_j^{\mathsf{p}} \cdot f_{ij} - \zeta_{ij}^{\mathsf{p}}} + \frac{1}{\mu_j \cdot f_{ij}}, \tag{10}$$

where ζ_{ij}^{p} denotes the arrival rate of instructions to node j for service i and measured as $\zeta_{ij}^{\mathsf{p}} = L_i^{\mathsf{p}} \cdot z_{ij} \cdot x_{ij}$, f_{ij} denotes the fraction of processing units that service i deployed on node j can obtain and is measured by $f_{ij} = \frac{L_i^{\mathsf{p}} \cdot \ x_{ij}}{\sum_{i=1}^{|A|} L_i^{\mathsf{p}} \cdot x_{ij}}$. $\mathrm{P}_{ij}^{\mathsf{Q}}$, referred to as Erlang's C formula, is measured as follows:

$$\mathrm{P}_{ij}^{\mathsf{Q}} = \frac{(n_j \cdot \rho_{ij})^{n_j}}{n_j!} . \frac{\mathrm{P}_{ij}^{\mathsf{0}}}{1 - \rho_{ij}}, \tag{11}$$

where $\rho_{ij} = \frac{\zeta_{ij}^{\mathsf{p}}}{F_j^{\mathsf{p}} \cdot f_{ij}}$ and $\mathrm{P}_{ij}^{\mathsf{0}}$ is calculated as follows:

$$\mathrm{P}_{ij}^{\mathsf{0}} = \left[\sum_{c=0}^{n_j - 1} \frac{(n_j \cdot \rho_{ij})^c}{c!} + \frac{(n_j \cdot \rho_{ij})^{n_j}}{n_j!} \left(\frac{1}{1 - \rho_{ij}} \right) \right]^{-1} \tag{12}$$

Finally, according to the QoS requirement, deadline violation is considered with respect to the percentage of requests from IoT services that their service delay exceed the delay threshold. We assume that desired quality for service[1] i is denoted by $\eta_i \,\epsilon\, (0,1)$ and the percentage of delay samples from services that exceed the delay threshold should be no more than $(1-\eta_i)$. We define V_{ij}^{m} as the percentage of requests of service i hosted on node j that do not meet the delay requirement during the time connected to the access point m. Then $V_i = \sum_{m=1}^{|M|} V_{ij}^{m} \cdot P_{im}$ measures the total violation percentage for the service. To conclude, Equation (13) calculates the cost of deadline violation for the placement plan δ:

$$O_{\delta}^{\mathsf{V}} = \sum_{i=1}^{|A|} \sum_{j=1}^{|C \bigcup F|} max(0, V_i - (1-\eta_i)) \cdot z_{ij} \cdot C_i^{\mathsf{v}} \cdot x_{ij} \cdot \tau \quad (13)$$

3) Cost of Service Deployment: Given the movements of end users, it is crucial for services to dynamically migrate across multiple nodes to maintain service performance and minimize user-perceived latency. However, frequent migration significantly increases operational costs. To address this trade-off, we investigate deployment cost, which measures the communication cost of service deployment from cloud nodes to fog nodes. Clients of service providers develop and upload new services to public cloud storage, and these services are then downloaded onto target nodes upon deployment. When the demand for a deployed service decreases, the host node may release the service to conserve space. If a fog node receives requests for a service not locally hosted, it must download and deploy the service locally. Notably, a cloud center has virtually unlimited storage space and can host services for an extended duration. As a result, the communication cost for service deployment on the cloud is omitted and Equation (14) measures the cost of service deployment.

$$O_{\delta}^{\mathsf{D}} = \sum_{i=1}^{|A|} \sum_{j=1}^{|F|} x_{ij} \cdot (1-\overline{x}_{ij}) \cdot L_i^{\mathsf{s}} \cdot C_{jc}^{\mathsf{c}}, \qquad (14)$$

where C_{jc}^{c} is the unit cost of communication between the fog node j and the cloud node c.

4) Cost of Communication: This cost encompasses data upload/download overhead, including service requests and responses transmission. Note that this component considers the costs of fog-to-fog and fog-to-cloud communication. In other words, the cost of communication between IoT and fog is not considered in this component, since this is usually outside the control of service providers. Services may be hosted on nodes different from those directly connected, requiring data transfer from the connecting access point to the host node. In addition, vehicle mobility may lead to traversing multiple access points during service offloading/processing. These factors contribute to fluctuating communication costs over defined time periods. In the illustrated example in Figure 1, the service requested by vehicle i is intended to be placed on fog node j. At the start of the time interval (t_0), i is linked to j through m, and by the end of the interval ($t_0+\tau$), it is connected to j through access point d. A relay mechanism, commonly referred to as "handoff" or "handover," allows a vehicle to obtain the service result from a neighboring access point if receiving it from the original access point is impractical. In other words, access points communicate with each other, and the service is relayed through the backhaul network [80], [81].

[1]Amazon compute SLA, available at https://aws.amazon.com/compute/sla (accessed April 2024).

Intermittent connectivity due to vehicle movement poses a significant risk to successful data transmission. Therefore, ensuring reliable link connectivity is crucial for the success of computation offloading. This reliability can be measured by the duration of link connections. Accordingly, Equation (15) measures the relevant communication cost.

$$O_{\delta}^{\mathsf{C}} = \sum_{i=1}^{|A|} \sum_{j=1}^{|C \bigcup F|} \sum_{m=1}^{|M|} x_{ij} \cdot (q_i + a_i) \cdot z_{ij} \cdot C_{mj}^{\mathsf{c}} \cdot \tau_{im}, \quad (15)$$

where the connectivity time τ_{im} represents the time during which vehicle i remains connected to access point m as $\tau_{im} = P_{im} \cdot \tau$ during the time interval τ.

5) Cost of Energy Consumption: The energy consumption of an IoT network comprises both static and dynamic components. The static part pertains to energy usage when resources are idle [82], while the dynamic component is determined by the current workload on active virtual machines and containers [14], [16]. As the static part remains unaffected by service placement policies, we focus solely on the dynamic component in our model. A significant portion of this component stems from computationally intensive requests [83], primarily attributed to networking and server machines.

$$O_{\delta}^{\mathsf{E}} = \tau \cdot (P_{\delta}^{\mathsf{n}} + P_{\delta}^{\mathsf{s}}) \qquad (16)$$

The dynamic power consumption of physical machines primarily depends on CPU utilization and can be modeled as a linear function [84]–[86] (refer to Equation (17)). Our fog servers, resembling nano data centers, are composed of single physical machines without additional ICT equipment such as fans and links [14]. To incorporate the energy consumption of additional ICT equipment, we utilize the Power Usage Effectiveness (PUE), denoted by θ_c, which signifies the ratio between total facility power consumption and IT equipment power consumption.

$$\begin{aligned} P_{\delta}^{s} \triangleq & \sum_{j=1}^{|F|} (P_j^{\mathsf{a}} - P_j^{\mathsf{i}}) \cdot u_j \cdot \left[(1-p_j^{\mathsf{r}}) \cdot C^{\mathsf{n}} + p_j^{\mathsf{r}} \cdot C^{\mathsf{r}}\right] \\ & \sum_{c=1}^{|C|} \theta_c \cdot (P_c^{\mathsf{a}} - P_c^{\mathsf{i}}) \cdot u_c \cdot \left[(1-p_c^{\mathsf{r}}) \cdot C^{\mathsf{n}} + p_c^{\mathsf{r}} \cdot C^{\mathsf{r}}\right], \end{aligned} \qquad (17)$$

where the parameter $(P_j^{\mathsf{a}} - P_j^{\mathsf{i}})/F_j^{\mathsf{p}}$ determines the incremental power consumption per unit load on the servers and $u_j = \frac{\zeta_j^p}{F_j^{\mathsf{p}}}$. To compute the dynamic energy consumption of the telecommunication network linking physical servers with vehicles we consider three main types of networking devices: edge routers, core routers, and switches. When a networking equipment carries traffic, it consumes load-dependent energy for packet processing and also for storing and forwarding the payload [87], [88]. Consequently, if the router j (the router connecting the fog server j to the network) receives z_{ij}

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

requests for service i during the time period τ, it processes $z_{ij} \cdot (a_i + q_i)$ bytes in such interval. Note that a cloud center is linked to the fog infrastructure through an edge router, whereas fog servers can be connected via either core routers or edge routers and access points. Additionally, mobile devices may connect to various access points within a given interval. Putting it all together, Equation (18) presents the networking power consumption of edge-to-cloud infrastructure as a result of the placement plan δ. The first segment in Equation (18) calculates the power consumption of networking devices (edge and core routers) connecting host nodes to the network. The subsequent part quantifies this metric for the switch within cloud centers, while the third segment assesses the energy consumed by access points connecting mobile devices to the network.

$$\begin{aligned} P^n_\delta = &\sum_{i=1}^{|A|}\sum_{j=1}^{|F|} x_{ij} \cdot z_{ij} \cdot p^{\mathsf{f}}_j \cdot (a_i + q_i) \cdot \\ &\left[(1 - p^{\mathsf{r}}_j) \cdot C^{\mathsf{n}} + p^{\mathsf{r}}_j \cdot C^{\mathsf{r}}\right] + \\ &\sum_{i=1}^{|A|}\sum_{c=1}^{|C|} \theta_c \cdot x_{ic} \cdot z_{ic} \cdot p^{\mathsf{f}}_{\mathsf{c}} \cdot (a_i + q_i) \cdot \\ &[(1 - p^{\mathsf{r}}_c) \cdot C^{\mathsf{n}} + p^{\mathsf{r}}_c \cdot C^{\mathsf{r}}] + \\ &\sum_{i=1}^{|A|}\sum_{j=1}^{|F \bigcup C|}\sum_{m=1}^{|M|} x_{ij} \cdot P_{im} \cdot z_{ij} \cdot p^{\mathsf{f}}_m \cdot (a_i + q_i) \cdot \\ &[(1 - p^{\mathsf{r}}_m) \cdot C^{\mathsf{n}} + p^{\mathsf{r}}_m \cdot C^{\mathsf{r}}] \end{aligned} \quad (18)$$

6) Cost of Carbon Footprint: According to the Shift Project report[1], carbon emission from information technology infrastructure and data servers supporting cloud computing now surpasses those from pre-Covid air travel. The carbon footprint is a critical consideration in server equipment deployment, and this study incorporates its cost into the optimization process, as shown in Equation (19). Here, R_c represents the average carbon emission rate for electricity per kilogram per kilowatt-hour [89]. The calculation only accounts for power consumption from non-renewable sources (P^E_δ), following the U.S. Energy Information Administration's classification of clean energy sources as carbon neutral.

$$O^{\mathsf{F}}_\delta = C^f \cdot R_c \cdot P^{\mathsf{E}}_\delta \cdot \tau, \quad (19)$$

C. Constraints

Resource utilization of fog nodes and cloud servers must not exceed their capacity, as formulated by:

$$\sum_{i=1}^{|A|} L^{\mathsf{s}}_i \cdot x_{ij} < F^{\mathsf{s}}_j, \text{ and} \sum_{i=1}^{|A|} L^{\mathsf{m}}_i \cdot x_{ij} < F^{\mathsf{m}}_j, \forall j \,\epsilon\, F \quad (20)$$

$$\sum_{i=1}^{|A|} L^{\mathsf{s}}_i \cdot x_{ic} < F^{\mathsf{s}}_c, \text{ and} \sum_{i=1}^{|A|} L^{\mathsf{m}}_i \cdot x_{ic} < F^{\mathsf{m}}_c, \forall c \,\epsilon\, C \quad (21)$$

[1]Environmental impact of digital: 5-year trends and 5G governance, available at https://theshiftproject.org/article/impact-environnemental-du-numerique-5g-nouvelle-etude-du-shift/ (accessed April 2024)

In addition, stability constraints of the queues for the services on fog nodes and cloud servers imply:

$$\zeta^{\mathsf{p}}_{ij} < F^{\mathsf{p}}_j \cdot f_{ij}, \text{ and } \zeta^{\mathsf{p}}_{ic} < F^{\mathsf{p}}_c \cdot f_{ic} \;\forall j \,\epsilon\, F, \forall c \,\epsilon\, C, \forall i \,\epsilon\, A \quad (22)$$

Finally, the placement of services is constrained so that each service must be hosted at most on one computational resource. Formally:

$$0 \le \sum_{i=1}^{|A|}\sum_{j=1}^{|C \bigcup F|} x_{ij} \le |A| \quad (23)$$

V. Cooperative Service Placement Algorithm

Achieving both system-wide and local objectives in our framework requires coordinated decision-making among agents, as local optimization alone cannot address the complexity of the problem. Agents must navigate multiple criteria, such as minimizing quadratic cost functions, to find optimal service placements. To handle this complexity, our placement algorithm autonomously balances these diverse objectives, relying on agents' local parameters to manage trade-offs. This section details how collaboration between fog agents enhances both system-wide and local performance.

The proposed distributed service placement algorithm tackles the problem periodically, avoiding bottlenecks from centralized servers and ensuring scalability in response to fluctuating resource demands. As illustrated in Figure 1, agents generate multiple service placement plans, evaluating them based on eight distinct cost components, and then collaborate in the plan selection phase to meet both local objectives (minimizing service provisioning costs) and global objectives (e.g., minimizing resource utilization variance). This approach extends previous research [31] by refining the methodology, incorporating multiple cost components, and addressing overarching system-wide objectives more comprehensively.

Initially, each fog agent autonomously generates multiple mappings (i.e., possible service placement plans) with different costs. In this step, the agents aim to minimize the service provisioning costs of their own received requests, thereby improving QoS and keeping service provisioning costs manageable. The possible plans encode selected hosts using a binary vector and resource utilization with a vector of real values [31]. The utilization vector reflects resource usage by representing the ratio of the assigned load to the capacity for each host node. This approach allows us to consider the heterogeneity in the capabilities of these nodes, providing a more nuanced representation of their individual capacities. As a result of the plan generation step, each agent calculates a set of potential alternative plans, each associated with the respective local cost.

To address the system-wide objectives, subsequently, all agents engage in collaborative decision-making to select one plan among their possible plans based on both local and global objectives. This research leverages EPOS [49], [50], a decentralized multi-agent system designed to tackle complex multi-objective discrete-choice combinatorial problems via a collective learning approach. It expands the scope of EPOS applicability by bridging computing and mobility, allowing for an examination of their interplay. Agents autonomously organize into a tree overlay topology to structure their interactions and facilitate information exchange [49], [50], thereby enabling cooperative optimization[1]. The optimization unfolds through a series of successive learning iterations, encompassing two phases: plan aggregation occurs in a bottom-up (leaves to root) fashion, while feedback propagation follows a top-down (root to leaves) approach within this structure. At each iteration t, agents refine their chosen plans by combining the two sets of local and global objectives in a weighted sum of costs, as depicted in Equation (24). This weighted summation facilitates making trade-offs and supports multiple levels of QoS.

$$\lambda \cdot L^t + (1 - \lambda) \cdot G^t, \tag{24}$$

where $\lambda \epsilon [0, 1]$. The higher the value of the weight, the stronger the preference towards minimizing the corresponding objective. The cost functions take as an argument the global plan (g) at the iteration $t - 1$, which is the sum of all utilization plans of the agents in the network. The global cost function and the local cost function are formulated as follows:

$$G^t = \sigma(g^t),\ L^t = min \frac{1}{|F|} \sum_{j=1}^{|F|} l(\delta_j^t), \tag{25}$$

where $G^t, L^t \epsilon \mathbb{R}$ and $l(.)$ extract the local cost of the selected plan δ of the agent j at iteration t and $|F|$ represents the number of fog agents engaged in the decision making.

MERA employs a local plan generation approach, which iterates for a specified number of plans. Consequently, its computational complexity is contingent upon on the number of iterations conducted within EPOS, with the critical path complexity linked to the network size. In EPOS, the tree height dictates the computational load, which scales logarithmically with the number of agents, denoted as $log|F|$. For each agent, the complexity is $O(|\delta| \cdot t)$. As a result, the overall complexity for all services is $O(|\delta| \cdot t \cdot \log |F|)$. In the subsequent section, we conduct novel experimental evaluations of the algorithm using real-world data.

VI. Evaluation

A fair comparison of the proposed cooperative placement approach of MERA with other state-of-the-art approaches is challenging given that the vast majority of related work relies on single controllers with centralized heuristics and machine learning approaches. Moreover, the state-of-the-art realism in the evaluation approach of MERA cannot be found in earlier approaches, which introduces significant threats for validity in the replication of the experimental conditions. Extensive comparisons of the collective learning approach of EPOS with other decentralized heuristics, such as the one of CO-HDA [90], [91], has been performed before, confirming the superior cost-effectiveness of EPOS as a state-of-the-art heuristic for discrete-choice combinatorial optimization problems of distributed multi-agent systems [49]. These comparisons demonstrate that EPOS can find better optimization solutions more efficiently, at a lower communication and computational cost [49]. Therefore, this paper focuses on the comparison of MERA with two *state-of-the-art localized approaches* in which the evaluation framework can be fully implemented for a fair comparison: *Baseline* and *Greedy*. Both comparisons are conducted across two distinct types of mobility profiles: *default* and *optimized* routes for vehicles, resulting in a total of six evaluation scenarios to thoroughly assess the effectiveness of the proposed framework.

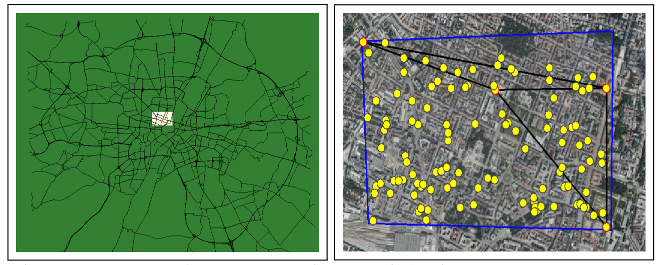

Fig. 2: Rectangular test area in Munich city center, core routers connected with black lines and LTE access points within the test area are highlighted.

In the Baseline approach, each fog server evaluates its own resource availability. If feasible, local deployment is chosen; otherwise, requests are forwarded to the cloud. The Greedy algorithm, referring to the widely-used First Fit approach in edge and cloud computing [24], [92], [93], aims to efficiently allocate resources while balancing optimization with computational efficiency. Upon receiving a request, a fog server assesses nearby resources with adequate capacity for the service, computes the provisioning cost, which comprises a total of eight local cost components, on each server, and sorts them based on ascending cost. It then selects the most suitable option from the beginning of the sorted list, iterating this process for subsequent requests. The Baseline approach focuses on offloading services solely to directly connected edge servers, aiming to minimize costs by utilizing local resources and a local perspective. Conversely, the Greedy approach pursues optimized solutions locally, leveraging a global view of the system.

The experiments are conducted for Munich, Germany. An area of $7000m^2$ of the city center is selected as the test area, shown in Figure. 2. A realistic traffic distribution pattern is derived from a real-world traffic trace representing the vehicle journeys within Munich city and the Sumo traffic simulator[2], employing two routing algorithms: (i) the Dijkstra algorithm, which determines the shortest paths for vehicles based on travel time and distance (default routes), and (ii) the duaIterate algorithm[3], which iteratively generates and refines the routes of vehicles to enhance traffic flow and alleviate congestion (optimized routes). Figure 3 shows the number of vehicles within the area and connected to the edge-to-cloud servers using the aforementioned routing algorithms over a three-hour period.

[1]The pseudo-code for the core EPOS algorithm is outlined in Algorithm 1 of the paper https://dl.acm.org/doi/pdf/10.1145/3277668.

[2]SUMO, available at https://sumo.dlr.de/docs/index.html (accessed April 2024).

[3]Duarouter, available at https://sumo.dlr.de/docs/duarouter.html (accessed April 2024).

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

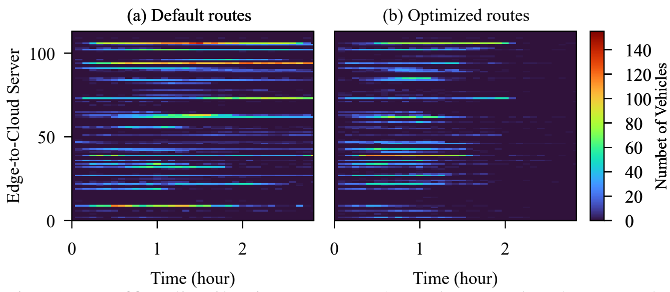

Fig. 3: Traffic distribution across the edge-to-cloud network: the number of connected vehicles to each access point, along with their presence time, is unbalanced and higher in default routes compared to optimized routes.

A. Simulation Setting

The incoming workload to the experiments originates form two sources: (i) vehicles traffic, illustrated in Figure 3, referred to as mobility profiles and (ii) IoT service requests, illustrated in Figure 4, referred to as IoT profiles. The IoT service requests come from the real-world traffic traces of Measurement and Analysis on the WIDE Internet (MAWI) traffic repository[1]. The MAWI archive, derived from the WIDE Internet backbone connecting Japanese academic and research institutions to the Internet, serves as a realistic dataset for modeling incoming IoT traffic rates to fog nodes [24]. Capturing daily traces at the transit link of WIDE to its upstream ISP, the archive includes packets covering common IoT protocols such as MQTT, COAP, mDNS, and AMQP that utilize TCP and UDP for message transmission. Figure 4 illustrates the average incoming traffic rate (z_{ij}) to the fog nodes per service within a 2-day time span. This traffic is subsequently distributed across the vehicles in the test area using a uniform distribution. Table II outlines additional characteristics of a mobile augmented reality application continuously running on each vehicle for autonomous driving [94], [95].

TABLE II: Experimentation parameters [24], [96]

Parameter	Value	Parameter	Value
Test Area	$2101.98 * 3313.97m^2$	L_i^{m}	U(2-400) MB
$\lvert C \rvert$	1	L_i^{p}	U(50,200) MI per req
$\lvert F \rvert$	114	L_i^{s}	U(50-500) MB
$\lvert S \rvert$	1	h_i	10 ms
$\lvert M \rvert$	109	q_i	U(10-26) KB
$\lvert A \rvert$	9788,10112	a_i	U(10-20) Byte
$\lvert N \rvert$	109	$\lvert \delta \rvert$	20
s_i	$\leq 120km/h$	θ_c	1.2[2]
τ	300 seconds	p_c^{r}	0.85[3]
λ	$\{0.05n \mid n \in \mathbb{N}_0, 0 \leq 0.05n \leq 1.0\}$	p_j^{f}	$\mathrm{Beta}(2, 0.5)$
C_j^{p}	edge 0.6, otherwise 0.2 USD per 1M req[4]	R_c	380 g CO2 eq/kWh[5]
C_j^{m}	$21 * 10^{-10}/128$ USD per MB per ms[6]	$b_{jj'}^{\mathsf{d}}$	[LTE: 0.072, Edge/Core: 10/100, Cloud: 100] Gbps
C_j^{s}	U(0.021,0.023) USD per GB per month[7]	$b_{jj'}^{\mathsf{u}}$	[LTE: 0.012, Edge/Core: 1/10, Cloud: 100] Gbps
$C_{jj'}^{\mathsf{c}}$	U(0.01,0.03) USD per GB (intra edge), U(0.03, 0.06) $ per GB (intra core or core-edge), U(0.06, 0.09) $ per GB (cloud communication)[8]	d_{ic}	U(15, 35) ms
C_i^{v}	[$\eta_i < 95.0\%$: 100%, $95.0\% < \eta_i < 99.0\%$: 30%, $\eta_i < 99.5\%$: 10%] Service Credit Percentage[9]	C_c	17.27 € per tonnes[10]
C^{n}	0.905 USD per KWh[11]	C^{r}	294 € per MWh[12]

The IoT workload is updated every 15 minutes based on the IoT trace data. Each 15-minute profile is divided into three 5-minute runs. Additionally, the number of vehicles in the test area is updated every 5 minutes. At the beginning of each 5-minute interval, (τ), the placement algorithms are executed, and the resulting placements are assumed to remain constant throughout the interval.

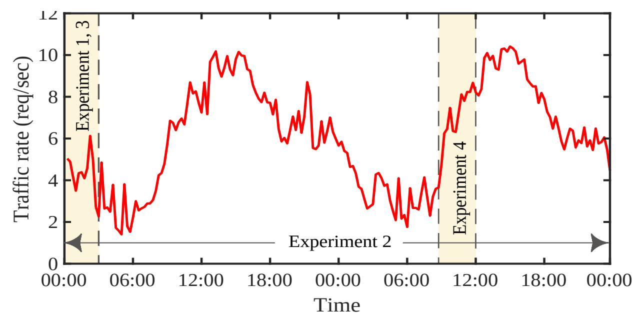

Fig. 4: 48 hours (192 15-minute IoT profiles) of MAWI trace data (April 12-13, 2017) serve as the input workload, delineating the experimental time intervals

1) Edge-to-Cloud Network: The experiments are conducted on the world's largest Open Database of Cell Towers (OpenCellID[13] dataset) that provides the locations of real-world cellular base stations. A total of 109 LTE Base Transceiver Stations situated within the Munich test area, as illustrated in Figure 2b. In the densely populated urban core of downtown Munich, each cell site is assumed to connect adjacent vehicles to the network within a coverage range of 700 meters, operating in the 2.6 GHz frequency band [97]. The network includes one cloud center located within a 10-hop distance from access points, with propagation delay U(15, 35) ms [24], and each cloud center requiring a Cisco Ethernet switch for operation [14]. The number and location of core routers within the test area is determined using the Macroscopic Internet Topology Caida dataset[14], EmuFog emulator [98], and the Munich Scientific Network (MWN)[15]. Tables II and IV outline the specification of the networking devices and their capacity.

The machines detailed in Table III exhibit comparable characteristics to the c5.metal instances introduced by Amazon

[1] MAWI working group traffic archive, available at http://mawi.wide.ad.jp/mawi (accessed April 2024).

[2] Cloud computing, server utilization, & the environment, available at https://aws.amazon.com/blogs/aws/cloud-computing-server-utilization-the-environment/ (accessed April 2024).

[3] International Energy Agency, available at https://www.iea.org/energy-system/buildings/data-centres-and-data-transmission-networks (accessed April 2024).

[4] AWS pricing, available at https://aws.amazon.com/lambda/pricing/ (accessed April 2024).

[5] Current emission in Germany, available at https://www.nowtricity.com/country/germany/ (accessed April 2024).

[6] AWS lambda pricing for a serverless application, available at https://www.clickittech.com/devops/aws-lambda-pricing/(accessed April 2024).

[7] AWS S3, available at https://aws.amazon.com/s3/pricing/(accessed April 2024).

[8] The guide to AWS data transfer pricing and saving, available at https://www.cloudbolt.io/guide-to-aws-cost-optimization/aws-data-transfer-pricing/ (accessed April 2024).

[9] AWS SLA, available at https://aws.amazon.com/compute/sla/ (accessed April 2024).

[10] Carbon taxes in Europe, available at https://taxfoundation.org/carbon-taxes-in-europe-2022/ (accessed April 2024).

[11] Germany electricity prices, available at https://www.globalpetrolprices.com/Germany/electricity_prices/ (accessed April 2024).

[12] S&P global commodity insights, available at https://www.spglobal.com/commodityinsights/en/market-insights/latest-news (accessed April 2024).

[13] OpenCelliD, available at https://opencellid.org (accessed April 2024).

[14] Macroscopic Internet topology, available at https://www.caida.org/data/internet-topology-data-kit/release-2019-04.xml (accessed April 2024).

[15] The Munich scientific network, available at https://www.lrz.de/services/netz/ (accessed April 2024).

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

TABLE III: Hardware specification and power consumption of the servers used in the evaluation

Machine model	Memory (GB)	Performance (GFLOPS)	Storage (GB)	Idle power (Watt)	Max power (Watt)
Xeon Gold 6140 36cores 2.3GHz	192	864	120	52.4	343
Xeon Gold 6136 24cores 3.0GHz	196	806.4	292	131	432
Xeon Platinum 8180 56cores 2.5GHz	192	1523.2	480	48	385
Xeon Platinum 8280 56cores 2.7GHz	192	1612.8	240	64.2	435
Xeon Platinum 8380HL 112cores 2.9GHz	384	1792	480	44.6	502
Cloud Xeon E5-2680 10cores 2.80 GHz	768	112000MIPS	500	57	115

TABLE IV: Networking equipment and their characteristics [87], [88], [96]

Device type	Idle power (Watt)	Max power (Watt)	Download max traffic capacity (Gbps)	Upload max traffic capacity (Gbps)	Download energy (Nj/bit)	Upload energy (Nj/bit)
3-sector 2x2 MIMO LTE base station	333	528	0.072	0.012	82820	12400
Cisco edge router 7609	4095	4550	560	560	37	37
Cisco core router CRS-3	11070	12300	4480	4480	12.6	12.6
Cisco Ethernet switch Catalyst 6509	1589	1766	256	256	31.7	31.7

in 2019[1], serving as our fog and cloud servers. Furthermore, it is assumed that the storage/processing capacity of the cloud center is sufficient to accommodate the generated load (unbounded). In 2020, renewable energy accounted for 50.9% of Germany's electricity generation[2]. Additionally, Munich, known for its focus on solar energy, aims to transition to a fully renewable energy system by 2025[3]. For our analysis, we assume that all the networking and computing resources, except for the cloud, are supplied by the same power utility, thereby sharing a consistent renewable energy profile. Specifically, we model the energy supplied from renewable sources for each access point and its co-located fog server using a Beta distribution with parameters $\alpha = 0.6$ and $\beta = 0.4$.

2) Costs: An important consideration is the time delay involved in deploying a service. If deploying services introduce extra delay, it could significantly impact the experienced service quality. Nevertheless, deploying containers typically take less than 50 ms [99], while the execution duration of the proposed framework for deploying services in practical scenarios could range from several seconds to minutes. In our simulations, we set the startup delay of service containers to 50 ms. Service costs are calculated by considering Amazon Web Services (AWS) Credits, which are determined as a percentage of monthly bill for Amazon EC2 instances. The instance chosen for analysis is t2.nano[4], featuring 1 vCPU, 0.5 GB of memory, and an On-Demand hourly credit of 0.0058 USD, aligning closely with the characteristics of the augmented reality service.

B. Results and Discussion

This section presents the results obtained from four distinct sets of experiments. The initial experiment investigates the delicate balance between service provisioning costs and workload equilibrium. The subsequent two experiments delve into the intricate interplay between costs and load balance within the realms of demand variability and resource limitations, respectively. The fourth experiment scrutinizes the correlation between costs and incentivization strategies toward renewable energy resources. Notably, excluding the third experiment, all fog nodes in the other three experiments operate at full capacity, with service utilization restricted to a maximum of 90%.

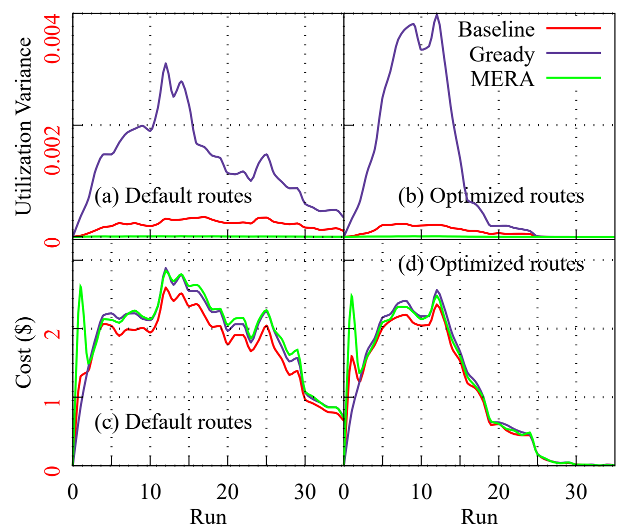

Fig. 5: The influence of traffic patterns on optimization objectives reveals distinct differences in workload distribution among placement approaches, while service provisioning costs are at a similar level.

1) Experiment 1: In this experiment, IoT profiles ranging from 0 to 11, as illustrated in Figure. 4, are distributed among the vehicles within the test area over a three-hour duration, comprising 36 5-min time slots. The association between optimization objectives (utilization variance and service provisioning cost, Figures 5 and 6) and the traffic pattern in the test area (Figure. 3) is evident across all approaches, indicating a clear influence of traffic load on the IoT workload and cost dynamics. Nonetheless, notable differences emerge in the workload distribution offered by different placement approaches.

Figure 5a and 5b display the overall variance in CPU and memory utilization, while Figure 7 illustrates the CPU utilization within the edge-to-cloud nodes. These figures collectively depict the load distribution across the network for the three tested approaches. Greedy demonstrates the highest variance

[1] Amazon EC2 C5 instances, available at https://aws.amazon.com/blogs/aws/now-available-new-c5-instance-sizes-and-bare-metal-instances/ (accessed April 2024).

[2] Renewables in Germany, available at https://en.wikipedia.org/wiki/Renewable_energy_in_GermanyStatistics (accessed April 2024).

[3] Sustainable business in Germany, available at https://www.reuters.com/business/sustainable-business/germany-aims-get-100-energy-renewable-sources-by-2035-2022-02-28/ (accessed April 2024).

[4] AWS EC2, available at https://aws.amazon.com/ec2/pricing/on-demand/ (accessed April 2024).

by concentrating the load on specific nodes. In contrast, the Baseline, which utilizes all nodes receiving requests, achieves a more well-distributed workload with lower variance. MERA, by spreading the workload evenly across a broader range of network nodes, offers the most effective balance. It is evident in the scenario of the default routes, both Baseline and Greedy exhibit considerable disparities compared to MERA. The Baseline approach, on average, shows a 40-fold increase in variance, while the Greedy approach displays an even much higher imbalance. In the scenario of optimized routes, while the increase of Baseline remains notable (28 times higher), the Greedy approach demonstrates even larger variance compared to MERA. This observation underscores the effective workload distribution of MERA, which overcome the significant distribution inefficiencies of the Baseline and Greedy approaches.

Figure 5c and 5d depict the cost of service provisioning across the three approaches, while Figure 6 dissects this cost into its constituent elements. In default scenarios, Baseline presents a modestly lower cost (10%) compared to MERA, while Greedy shows a slight advantage (3%) over MERA. However, in optimized route scenarios, these differences diminish, with Baseline demonstrating 6% and Greedy 1% lower cost compared to MERA. In the optimized scenario, the costs decrease by 45%, 42%, and 44% for MERA, Baseline, and Greedy, respectively. This emphasizes the significance of uncongested traffic networks in reducing costs within ICT networks and enabling more efficient resource utilization.

Across both optimized and default route scenarios, the differences in storage, memory, energy, and CO_2 costs between Baseline and Greedy approaches compared to MERA are lower than 2.4%, with negligible variations observed. Initially, MERA incurs the highest deployment cost due to distributing services across the network, while the Baseline approach experiences a smaller yet significant cost by locally deploying services on all receivers, resulting in a decrease ranging from 37% to 48% compared to MERA. Conversely, the Greedy approach, which deploys services on a few specific nodes, exhibits a comparatively lower cost, with a decrease of 97% compared to MERA. The cost of deadline violation is lower in MERA, as it balances service distribution across the network, ensuring adequate processing power for each service. In contrast, the utilization of only a few nodes in Greedy leads to higher queuing and execution delays for services. Additionally, the distribution of services closer to the origins in MERA mitigates communication delays. The Baseline approach incurs a deadline cost approximately 14% to 17% higher than MERA, while the Greedy approach exhibits a more substantial increase, ranging from 31% to 34%. This underscores the effectiveness of MERA, particularly in time-sensitive applications. The offloading of services across the network results in higher communication cost for both MERA and Greedy compared to Baseline. The MERA and Greedy approaches demonstrate a similar increase of 3% compared to Baseline in the optimized scenario, while the difference ranges from 8% to 9% in the default scenario. Table V in Appendix A offers a more detailed comparison of the costs associated with the Baseline and Greedy approaches compared to MERA.

The Greedy approach highlights the tendency of agents to prioritize their individual objective without coordinating towards the system-wide goal, leading to striking fragmentation over time as certain nodes handle a significant workload, while others remain underutilized. Consequently, this fragmentation leads to overloaded nodes struggling to maintain high-quality service standards, resulting in increased deadline violations and degraded overall system performance. Although the Baseline provides the lowest placement cost, it results in a better load distribution yet imbalanced. Thus, the Greedy and Baseline approaches' lack of coordination can exacerbate disparities in workload distribution, adversely impacting the system's efficiency and reliability. In contrast, MERA achieves a balanced workload at a slightly higher provisioning cost compared to the Baseline approach. However, this additional cost can be effectively mitigated by caching or storing frequently used services locally by service providers.

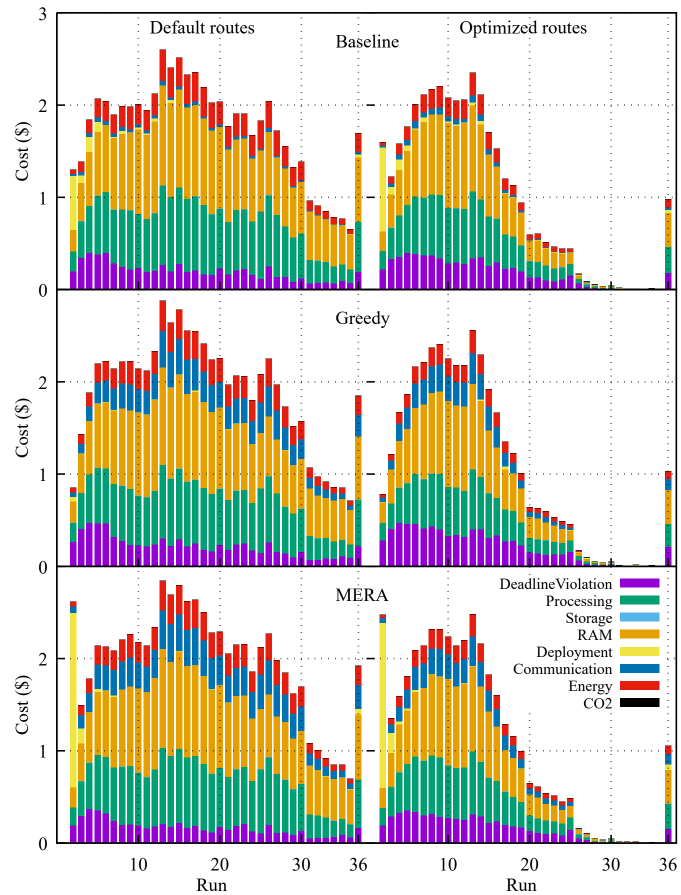

Fig. 6: MERA outperforms other approaches in deadline violation, Baseline excels in communication cost, and Greedy offers the lowest deployment cost, with other costs being nearly similar. Run 36 denotes the mean value of costs across all runs for each approach.

2) *Experiment 2:* In this experiment, a 3-hour window is initially defined, encompassing 12 consecutive 5-minute IoT profiles. Subsequently, the experiments are executed for all windows, totaling 179 windows. Figure. 8 presents the average of IoT traffic across the windows alongside the corresponding results, highlighting the influence of traffic fluctuations on the service placement goals. As IoT load decreases, all placement approaches show lower and more closely aligned execution

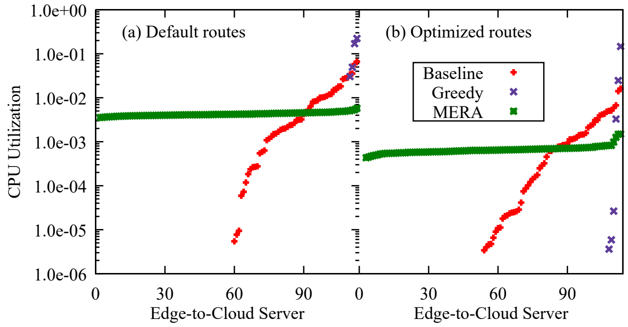

Fig. 7: Greedy concentrates load on specific nodes, while Baseline distributes workload across all receivers. MERA achieves the most balanced workload distribution. Servers are sorted by utilization.

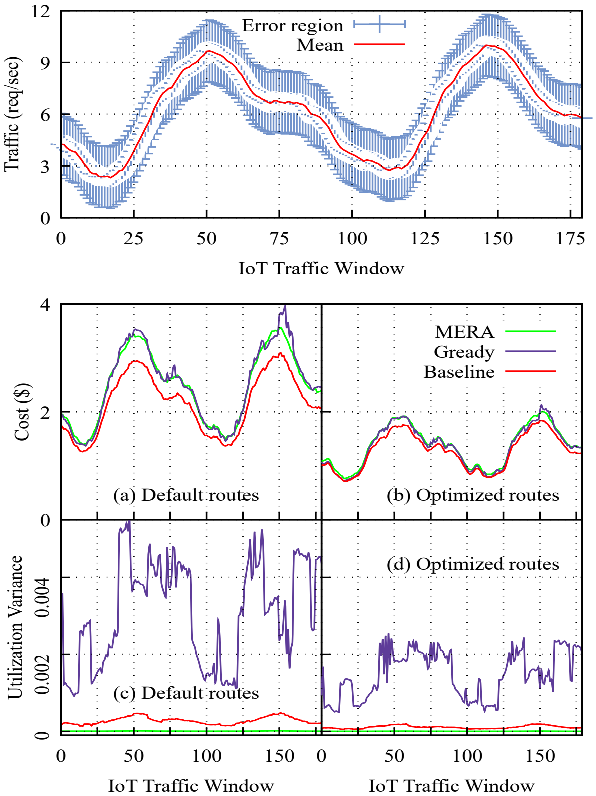

Fig. 8: Traffic variation across 12-profile IoT traffic windows influences load balance and service provisioning cost. Approaches diverge notably during traffic peaks, with Greedy exhibiting the largest fluctuations

costs. Conversely, with increasing load, costs diverge, especially noticeable in default route scenarios imposing heavier load on the ICT network. At the peak IoT load (window 154), the Greedy approach incurs 34% higher costs compared to the Baseline, which decreases to 15% in optimized routes. Comparing the cost at the peak load between MERA and the Baseline shows a 10% increase in MERA under optimized conditions and a 15% increase in the default scenario. Consequently, MERA demonstrates greater scalability and robustness to IoT traffic peaks compared to Greedy in terms of service provisioning cost.

On average, transitioning from default to optimized routes results in utilization variance reductions of 46%, 60%, and 51% for MERA, Baseline, and Greedy, respectively. This highlights a significant decrease in variability achieved with optimized traffic configurations. Regarding service provisioning cost, in the default routes scenario, MERA, Baseline, and Greedy approaches exhibit average placement costs that are 44%, 41%, and 46% higher, respectively, compared to their counterparts using optimized routes. This underscores the notable reduction in costs associated with optimized traffic configurations. This discrepancy arises from the more even distribution of vehicles across the test area and quicker departure of traffic from the city via optimized routes, reducing the load on the ICT network over a shorter duration compared to default routes.

3) Experiment 3: This experiment assesses how network capacity and resource limitations influence service distribution and service provisioning costs. The proposed approach systematically adjusts the available capacity, ranging from 80% to 120% of the current demand in each run. This capacity is then allocated across a specific number of nodes (randomly selected) within the fog infrastructure, ranging from 10 to 110 nodes in increments of 5. Each experiment is repeated 10 times to ensure robustness, with each data point on the Figure 9 and 11 representing the average outcome derived from these 10 samples of network nodes for default and optimized routes scenarios, respectively.

In the default scenario, the Greedy approach exhibits fluctuations and a rising trend as the number of active nodes increases across various capacity/demand ratios. This suggests that, despite similar capacity relative to demand, the Greedy approach faces challenges in resource allocation efficiency. Moreover, this variability intensifies with the increase in the number of nodes, as the Greedy approach opts to deploy services on a limited number of nodes among the available options. The Baseline initially shows an upward trend before stabilizing, indicating a more adaptive resource distribution compared to the Greedy approach. This leads to a more consistent server utilization pattern over time for the Baseline approach. Although fluctuations emerge with increased network capacity, possibly due to the inherent complexity of load balancing in edge-to-cloud networks, the MERA approach demonstrates a smooth decreasing trend, indicating the presence of robust load management strategies that effectively adapt to evolving network conditions.

The increase in fog network capacity reduces service placement costs for all approaches, benefiting both service providers and users. While the Greedy and Baseline approaches exhibit a consistent close trend across variations in the number of nodes and capacity-to-demand ratio, MERA experiences a gradual growth when the number of active nodes reaches the range of 40 to 60. In the worst-case scenario (capacity-to-demand ratio of 0.8), the cost increase can reach up to 25% compared to Greedy and Baseline. However, this increase diminishes as the capacity-to-demand ratio increases and even falls below that of Greedy when the nodes are operating at full capacity, see Figures. 5 (c,d), 8 (a,b), and 10 (c,d). This suggests that distributing the workload across more nodes may lead to adverse effects on costs, particularly due to communication

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

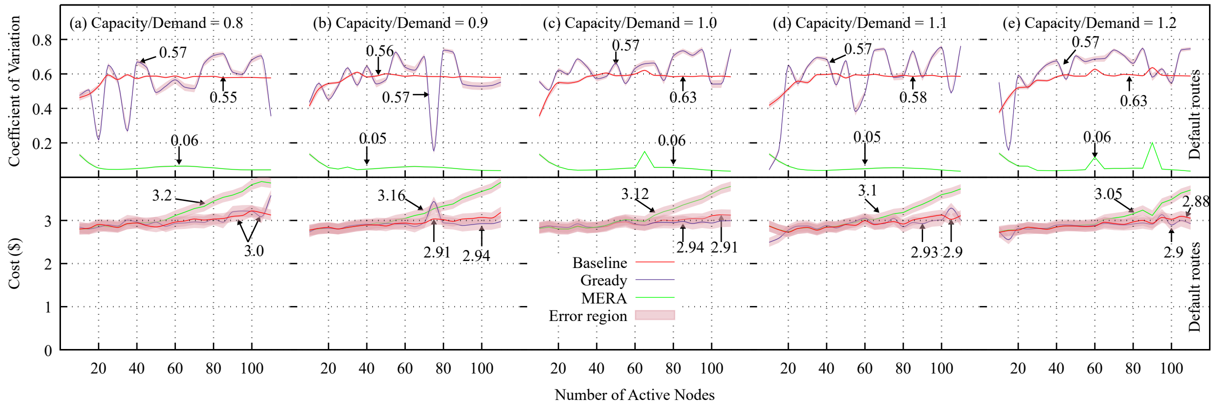

Fig. 9: The influence of edge-to-cloud capacity/size on servers' utilization dispersion and service provisioning cost: Increasing capacity reduces costs, while a rise in the number of active nodes decreases utilization variation in MERA but increases it in other methods. Labels depict the mean coefficients of variation and costs for each approach.

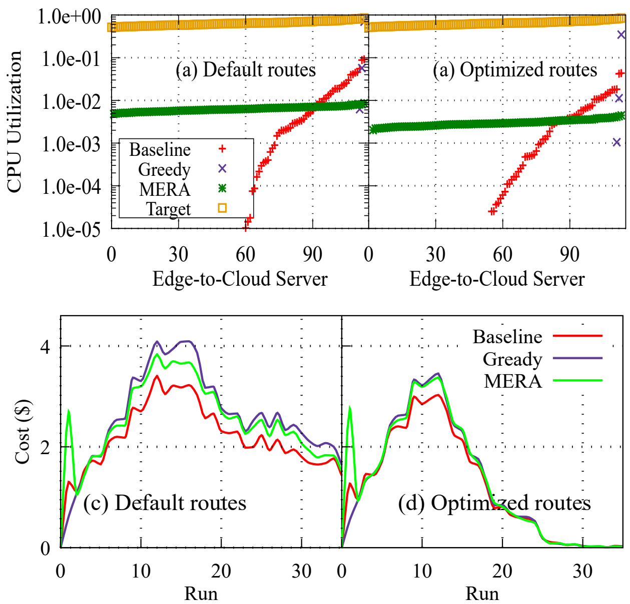

Fig. 10: MERA loads nodes in accordance with their renewable energy penetration rate, while Baseline deploys services locally, sacrificing load alignment for cost reduction. Greedy, on the other hand, fails to achieve superiority in either objective.

and deployment expenses.

4) Experiment 4: This experiment demonstrates how MERA prioritizes renewable energy sources over non-renewable powered servers, as illustrated by the logarithmic trends depicted in Figure 10a and 10b. MERA optimizes for this by minimizing the RMSE between the utilization and renewable power ratio of each server across the entire network. In this figure, the orange points represent the renewable power ratio of each server, while the other points depict the utilization of those servers across different approaches. The servers are sorted by utilization.

The upward trend observed in the MERA approach indicates that as the renewable power ratio increases, so does the utilization of the servers. This trend highlights the effectiveness of prioritizing renewable energy sources, leading to higher utilization rates. Conversely, the Baseline and Greedy approaches prioritize local deployment and minimize service provisioning costs, respectively, which lead to deviations from the target signal in these approaches.

Given the upward trend in incoming IoT traffic depicted in Figure 4 for this experiment, Figure 10c and 10d show a notably higher provisioning cost compared to Experiment 1, where the IoT traffic remained relatively stable. Specifically, in the default scenario, we observe an increase of 29%, 27%, and 39% for MERA, Baseline, and Greedy, respectively. In contrast, in the optimized scenario, the increases are relatively lower, at 22%, 19%, and 19% for MERA, Baseline, and Greedy, respectively.

VII. Conclusion

This paper aims to pave the way for a more efficient and sustainable smart mobility and computing ecosystem. Given the transformative potential of ITS in fostering safer and more eco-friendly transportation systems, the increasing proliferation of IoT devices within vehicles, coupled with their significant energy demands and processing costs in the context of edge-to-cloud infrastructure, poses a formidable challenge to the readiness of ICT. To address these challenges, this paper introduces a pioneering distributed service placement framework designed to dynamically manage the load of smart mobility services within the edge-to-cloud ecosystem. By optimizing QoS, service provisioning costs, workload balance, and sustainability considerations, this framework offers a novel approach to cope with the dynamic and stochastic nature of vehicular networks. Through extensive evaluation based on real-world infrastructure settings and traffic traces from Munich, the proposed framework demonstrates the potential to enhance computing resilience and empower self-adaptive ICT infrastructures.

To further advance the scope and impact of this research, several avenues for future work have been identified. The framework could be extended to encompass additional dynamic edge computing applications, such as drone-based systems [100]. Additionally, conducting experiments in real-world testbeds, such as the SMOTEC platform [58], could provide valuable validation and refinement of the proposed framework. Furthermore, further study into multi-modal traffic patterns, particularly those associated with the transition to low-carbon transport modalities, holds promise for yielding valuable insights into the interactions between mobility dynamics and computing resources.

VIII. Acknowledgments

This project is funded by a UKRI Future Leaders Fellowship (MR-/W009560-/1): '*Digitally Assisted Collective Governance of Smart City Commons–ARTIO*'

References

[1] Jiacheng Chen, Haibo Zhou, Ning Zhang, Wenchao Xu, Quan Yu, Lin Gui, and Xuemin Shen. Service-oriented dynamic connection management for software-defined internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 18(10):2826–2837, 2017.

[2] Alexandre K Ligo, Jon M Peha, Pedro Ferreira, and João Barros. Throughput and economics of dsrc-based internet of vehicles. *IEEE Access*, 6:7276–7290, 2017.

[3] Ning Lu, Nan Cheng, Ning Zhang, Xuemin Shen, and Jon W Mark. Connected vehicles: Solutions and challenges. *IEEE internet of things journal*, 1(4):289–299, 2014.

[4] Chun-Cheng Lin, Der-Jiunn Deng, and Chia-Chi Yao. Resource allocation in vehicular cloud computing systems with heterogeneous vehicles and roadside units. *IEEE Internet of Things Journal*, 5(5):3692–3700, 2017.

[5] David Reinsel-John Gantz-John Rydning, John Reinsel, and John Gantz. The digitization of the world from edge to core. *Framingham: International Data Corporation*, 16:1–28, 2018.

[6] Nicola Jones. How to stop data centres from gobbling up the world's electricity. *Nature*, 561(7722):163–167, 2018.

[7] Zhuo Chen, Wenlu Hu, Junjue Wang, Siyan Zhao, Brandon Amos, Guanhang Wu, Kiryong Ha, Khalid Elgazzar, Padmanabhan Pillai, Roberta Klatzky, et al. An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance. In *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, pages 1–14, 2017.

[8] Flavio Bonomi, Rodolfo Milito, Preethi Natarajan, and Jiang Zhu. Fog computing: A platform for internet of things and analytics. In *Big data and internet of things: A roadmap for smart environments*, pages 169–186. Springer, 2014.

[9] Qiuping Li, Junhui Zhao, Yi Gong, and Qingmiao Zhang. Energy-efficient computation offloading and resource allocation in fog computing for internet of everything. *China Communications*, 16(3):32–41, 2019.

[10] Zeinab Nezami and Kamran Zamanifar. Internet of things/internet of everything: structure and ingredients. *IEEE Potentials*, 38(2):12–17, March 2019.

[11] Dimas Satria, Daihee Park, and Minho Jo. Recovery for overloaded mobile edge computing. *Future Generation Computer Systems*, 70:138–147, 2017.

[12] Ilias Gerostathopoulos and Evangelos Pournaras. Trapped in traffic? a self-adaptive framework for decentralized traffic optimization. In *14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 32–38. IEEE/ACM, 2019.

[13] Brionna Davis, Grace Jennings, Taylor Pothast, Ilias Gerostathopoulos, Evangelos Pournaras, and Raphael E Stern. Decentralized optimization of vehicle route planning—a cross-city comparative study. *IEEE Internet Computing*, 25(3):34–42, 2021.

[14] Ehsan Ahvar, Anne-Cécile Orgerie, and Adrien Lebre. Estimating energy consumption of cloud, fog and edge computing infrastructures. *IEEE Transactions on Sustainable Computing*, 2019.

[15] Ming Yan, Chien Aun Chan, André F Gygax, Jinyao Yan, Leith Campbell, Ampalavanapillai Nirmalathas, and Christopher Leckie. Modeling the total energy consumption of mobile network services and applications. *Energies*, 12(1):184, 2019.

[16] Farah Ait Salaht, Frédéric Desprez, and Adrien Lebre. An overview of service placement problem in fog and edge computing. *ACM Computing Surveys (CSUR)*, 53(3):1–35, 2020.

[17] Ashkan Yousefpour, Caleb Fung, Tam Nguyen, Krishna Kadiyala, Fatemeh Jalali, Amirreza Niakanlahiji, Jian Kong, and Jason P Jue. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, 2019.

[18] Parviz Keshavarz Haddadha, Mohammad Hossein Rezvani, Mahdi MollaMotalebi, and Achyut Shankar. Machine learning methods for service placement: a systematic review. *Artificial Intelligence Review*, 57(3):61, 2024.

[19] Fatemeh Sarkohaki and Mohsen Sharifi. Service placement in fog–cloud computing environments: a comprehensive literature review. *The Journal of Supercomputing*, pages 1–33, 2024.

[20] Yueyue Dai, Du Xu, Sabita Maharjan, and Yan Zhang. Joint load balancing and offloading in vehicular edge computing and networks. *IEEE Internet of Things Journal*, 6(3):4377–4387, 2018.

[21] Dadmehr Rahbari and Mohsen Nickray. Task offloading in mobile fog computing by classification and regression tree. *Peer-to-Peer Networking and Applications*, 13:104–122, 2020.

[22] Tuyen X Tran and Dario Pompili. Joint task offloading and resource allocation for multi-server mobile-edge computing networks. *IEEE Transactions on Vehicular Technology*, 68(1):856–868, 2018.

[23] Dragi Kimovski, Narges Mehran, Christopher Emanuel Kerth, and Radu Prodan. Mobility-aware iot applications placement in the cloud edge continuum. *IEEE Transactions on Services Computing*, 2021.

[24] Ashkan Yousefpour, Ashish Patil, Genya Ishigaki, Inwoong Kim, Xi Wang, Hakki C Cankaya, Qiong Zhang, Weisheng Xie, and Jason P Jue. Fogplan: A lightweight qos-aware dynamic fog service provisioning framework. *IEEE Internet of Things Journal*, 2019.

[25] Carla Mouradian, Somayeh Kianpisheh, Mohammad Abu-Lebdeh, Fereshteh Ebrahimnezhad, Narjes Tahghigh Jahromi, and Roch H Glitho. Application component placement in nfv-based hybrid cloud/fog systems with mobile fog nodes. *IEEE Journal on Selected Areas in Communications*, 37(5):1130–1143, 2019.

[26] Paridhika Kayal and Jörg Liebeherr. Autonomic service placement in fog computing. In *2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, pages 1–9. IEEE, 2019.

[27] Yunpeng Wang, Ping Lang, Daxin Tian, Jianshan Zhou, Xuting Duan, Yue Cao, and Dezong Zhao. A game-based computation offloading method in vehicular multiaccess edge computing networks. *IEEE Internet of Things Journal*, 7(6):4987–4996, 2020.

[28] Tao Ouyang, Zhi Zhou, and Xu Chen. Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing. *IEEE Journal on Selected Areas in Communications*, 36(10):2333–2345, 2018.

[29] Farhad Tavousi, Sadoon Azizi, and Abdulbaghi Ghaderzadeh. A fuzzy approach for optimal placement of iot applications in fog-cloud computing. *Cluster Computing*, pages 1–18, 2022.

[30] John Paul Martin, A Kandasamy, and K Chandrasekaran. Mobility aware autonomic approach for the migration of application modules in fog computing environment. *Journal of Ambient Intelligence and Humanized Computing*, 11:5259–5278, 2020.

[31] Zeinab Nezami, Kamran Zamanifar, Karim Djemame, and Evangelos Pournaras. Decentralized edge-to-cloud load balancing: Service placement for the internet of things. *IEEE Access*, 9:64983–65000, 2021.

[32] Khaled M Matrouk and Amer D Matrouk. Mobility aware-task scheduling and virtual fog for offloading in iot-fog-cloud environment. *Wireless Personal Communications*, 130(2):801–836, 2023.

[33] Tanissia Djemai, Patricia Stolf, Thierry Monteil, and Jean-Marc Pierson. Investigating mobility-aware strategies for iot services placement in the fog under energy and qos constraints. *Journal of Communications Software and Systems*, 17(2):73–86, 2021.

[34] Tanissia Djemai, Patricia Stolf, Thierry Monteil, and Jean-Marc Pierson. Mobility support for energy and qos aware iot services placement in the fog. In *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–7. IEEE, 2020.

[35] Tayebeh Bahreini, Marco Brocanelli, and Daniel Grosu. Vecman: A framework for energy-aware resource management in vehicular edge computing systems. *IEEE Transactions on Mobile Computing*, 2021.

[36] Tanissia Djemai, Patricia Stolf, Thierry Monteil, and Jean-Marc Pierson. A discrete particle swarm optimization approach for energy-efficient iot services placement over fog infrastructures. In *2019 18th International Symposium on Parallel and Distributed Computing (ISPDC)*, pages 32–40. IEEE, 2019.

[37] Lei Liu, Jie Feng, Xuanyu Mu, Qingqi Pei, Dapeng Lan, and Ming Xiao. Asynchronous deep reinforcement learning for collaborative task computing and on-demand resource allocation in vehicular edge computing. *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[38] Hongjian Li, Chen Xu, Tiantian Wang, Jingjing Wang, Peng Zheng, Tongming Liu, and Libo Tang. A cost-efficient and QoS-aware adaptive placement of applications in fog computing. *Concurrency and Computation: Practice and Experience*, page e7701, 2023.

[39] Daniel Happ, Suzan Bayhan, and Vlado Handziski. Joi: Joint placement of IoT analytics operators and pub/sub message brokers in fog-centric IoT platforms. *Future generation computer systems*, 119:7–19, 2021.

[40] Zoltán Adám Mann. Secure software placement and configuration. *Future Generation Computer Systems*, 110:243–253, 2020.

[41] Zheyi Chen, Junqin Hu, Xing Chen, Jia Hu, Xianghan Zheng, and Geyong Min. Computation offloading and task scheduling for dnn-based applications in cloud-edge computing. *IEEE Access*, 8:115537–115547, 2020.

[42] Wenhao Fan, Jie Liu, Mingyu Hua, Fan Wu, and Yuan'an Liu. Joint task offloading and resource allocation for multi-access edge computing assisted by parked and moving vehicles. *IEEE Transactions on Vehicular Technology*, 71(5):5314–5330, 2022.

[43] Chaogang Tang, Chunsheng Zhu, Huaming Wu, Qing Li, and Joel JPC Rodrigues. Toward response time minimization considering energy consumption in caching-assisted vehicular edge computing. *IEEE Internet of Things Journal*, 9(7):5051–5064, 2021.

[44] Xiaoyu Zhu, Yueyi Luo, Anfeng Liu, Md Zakirul Alam Bhuiyan, and Shaobo Zhang. Multiagent deep reinforcement learning for vehicular computation offloading in iot. *IEEE Internet of Things Journal*, 8(12):9763–9773, 2020.

[45] Hamza Sulimani, Akbar Muhammad Sajjad, Wael Y Alghamdi, Omprakash Kaiwartya, Tony Jan, Simeon Simoff, and Mukesh Prasad. Reinforcement optimization for decentralized service placement policy in iot-centric fog environment. *Transactions on Emerging Telecommunications Technologies*, 34(11):e4650, 2023.

[46] Uchechukwu Awada, Jiankang Zhang, Sheng Chen, Shuangzhi Li, and Shouyi Yang. Resource-aware multi-task offloading and dependency-aware scheduling

for integrated edge-enabled IoV. *Journal of Systems Architecture*, page 102923, 2023.

[47] Wenhan Zhan, Chunbo Luo, Jin Wang, Chao Wang, Geyong Min, Hancong Duan, and Qingxin Zhu. Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing. *IEEE Internet of Things Journal*, 7(6):5449–5465, 2020.

[48] Bo Li, Qiang He, Guangming Cui, Xiaoyu Xia, Feifei Chen, Hai Jin, and Yun Yang. Read: Robustness-oriented edge application deployment in edge computing environment. *IEEE Transactions on Services Computing*, 15(3):1746–1759, 2020.

[49] Evangelos Pournaras, Peter Pilgerstorfer, and Thomas Asikis. Decentralized collective learning for self-managed sharing economies. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 13(2):10, 2018.

[50] Evangelos Pournaras. Collective learning: A 10-year odyssey to human-centered distributed intelligence. In *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*, pages 205–214. IEEE, 2020.

[51] Lotfi Abdi, Faten Ben Abdallah, and Aref Meddeb. In-vehicle augmented reality traffic information system: a new type of communication between driver and vehicle. *Procedia Computer Science*, 73:242–249, 2015.

[52] BB Prahlada Rao, Paval Saluia, Neetu Sharma, Ankit Mittal, and Shivay Veer Sharma. Cloud computing for internet of things & sensing based applications. In *2012 Sixth International Conference on Sensing Technology (ICST)*, pages 374–380. IEEE, 2012.

[53] Sadoon Azizi, Fariba Khosroabadi, and Mohammad Shojafar. A priority-based service placement policy for fog-cloud computing systems. *Computational Methods for Differential Equations*, 7(4):521–534, 2019.

[54] Harshit Gupta, Amir Vahid Dastjerdi, Soumya K Ghosh, and Rajkumar Buyya. iFogSim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

[55] Lilian Hernandez, Hung Cao, and Monica Wachowicz. Implementing an edge-fog-cloud architecture for stream data management. In *2017 IEEE Fog World Congress (FWC)*, pages 1–6. IEEE, 2017.

[56] Yunbo Li, Anne-Cécile Orgerie, Ivan Rodero, Betsegaw Lemma Amersho, Manish Parashar, and Jean-Marc Menaud. End-to-end energy models for edge cloud-based IoT platforms: Application to data stream analysis in iot. *Future Generation Computer Systems*, 87:667–678, 2018.

[57] Francesco Malandrino, Claudio Casetti, Carla-Fabiana Chiasserini, and Marco Fiore. Content download in vehicular networks in presence of noisy mobility prediction. *IEEE Transactions on Mobile Computing*, 13(5):1007–1021, 2013.

[58] Zeinab Nezami, Evangelos Pournaras, Amir Borzouie, and Jie Xu. Smotec: An edge computing testbed for adaptive smart mobility experimentation. In *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pages 1–7. IEEE, 2023.

[59] Shahid H. Bokhari. On the mapping problem. *IEEE Trans. Computers*, 30(3):207–214, 1981.

[60] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. Profit-aware resource management for edge computing systems. In *Proceedings of the 1st International Workshop on Edge Systems, Analytics and Networking*, pages 25–30, 2018.

[61] Jinlai Xu, Balaji Palanisamy, Heiko Ludwig, and Qingyang Wang. Zenith: Utility-aware resource allocation for edge computing. In *Edge Computing (EDGE), 2017 IEEE International Conference on*, pages 47–54. IEEE, 2017.

[62] Daewoo Kim, Hyojung Lee, Hyungseok Song, Nakjung Choi, and Yung Yi. Economics of fog computing: interplay among infrastructure and service providers, users, and edge resource owners. *IEEE Transactions on Mobile Computing*, 19(11):2609–2622, 2019.

[63] Arash Bozorgchenani, Setareh Maghsudi, Daniele Tarchi, and Ekram Hossain. Computation offloading in heterogeneous vehicular edge networks: On-line and off-policy bandit solutions. *IEEE Transactions on Mobile Computing*, 2021.

[64] Hyung Gyu Lee and Naehyuck Chang. Powering the iot: Storage-less and converter-less energy harvesting. In *The 20th Asia and South Pacific Design Automation Conference*, pages 124–129. IEEE, 2015.

[65] Hasan Ali Khattak, Hafsa Arshad, Saif ul Islam, Ghufran Ahmed, Sohail Jabbar, Abdullahi Mohamud Sharif, and Shehzad Khalid. Utilization and load balancing in fog servers for health applications. *EURASIP Journal on Wireless Communications and Networking*, 2019(1):91, 2019.

[66] Amir-Mohammad Rahmani, Nanda Kumar Thanigaivelan, Tuan Nguyen Gia, Jose Granados, Behailu Negash, Pasi Liljeberg, and Hannu Tenhunen. Smart e-health gateway: Bringing intelligence to internet-of-things based ubiquitous healthcare systems. In *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, pages 826–834. IEEE, 2015.

[67] Tuan Nguyen Gia and Mingzhe Jiang. Exploiting fog computing in health monitoring. *Fog and Edge Computing: Principles and Paradigms*, pages 291–318, 2019.

[68] Davide Andrea Guastella and Evangelos Pournaras. Cooperative multi-agent traffic monitoring can reduce camera surveillance. *IEEE Access*, 2023.

[69] José Suárez-Varela and Pere Barlet-Ros. Sbar: SDN flow-based monitoring and application recognition. In *Proceedings of the Symposium on SDN Research*, pages 1–2, 2018.

[70] Duc-Hung Luong, Abdelkader Outtagarts, and Abdelkrim Hebbar. Traffic monitoring in software defined networks using opendaylight controller. In *Mobile, Secure, and Programmable Networking: Second International Conference, MSPN 2016, Paris, France, June 1-3, 2016, Revised Selected Papers 2*, pages 38–48. Springer, 2016.

[71] Wei Li, Ting Yang, Flavia C Delicato, Paulo F Pires, Zahir Tari, Samee U Khan, and Albert Y Zomaya. On enabling sustainable edge computing with renewable energy resources. *IEEE Communications Magazine*, 56(5):94–101, 2018.

[72] Yashar Ghiassi-Farrokhfal, Srinivasan Keshav, and Catherine Rosenberg. Toward a realistic performance analysis of storage systems in smart grids. *IEEE Transactions on Smart Grid*, 6(1):402–410, 2014.

[73] Mohammad Mainul Islam, Fahimeh Ramezani, Hai Yan Lu, and Mohsen Naderpour. Optimal placement of applications in the fog environment: A systematic literature review. *Journal of Parallel and Distributed Computing*, 174:46–69, 2023.

[74] Quang Duy La, Mao V Ngo, Thinh Quang Dinh, Tony QS Quek, and Hyundong Shin. Enabling intelligence in fog computing to achieve energy and latency reduction. *Digital Communications and Networks*, 5(1):3–9, 2019.

[75] Ruilong Deng, Rongxing Lu, Chengzhe Lai, Tom H Luan, and Hao Liang. Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. *IEEE Internet of Things Journal*, 3(6):1171–1181, 2016.

[76] Carlo Puliafito, Diogo M Gonçalves, Márcio M Lopes, Leonardo L Martins, Edmundo Madeira, Enzo Mingozzi, Omer Rana, and Luiz F Bittencourt. Mobfogsim: Simulation of mobility and migration for fog computing. *Simulation Modelling Practice and Theory*, 101:102062, 2020.

[77] Yasir Saleem, Nathalie Mitton, and Valeria Loscri. A vehicle-to-infrastructure data offloading scheme for vehicular networks with QoS provisioning. In *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pages 1442–1447. IEEE, 2021.

[78] Natarajan Gautam. *Analysis of Queues*. CRC Press, LLC, Boca Raton, Florida, United States, 2012.

[79] Gunter Bolch, Stefan Greiner, Hermann De Meer, and Kishor S Trivedi. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications*. John Wiley & Sons, 2006.

[80] Ke Zhang, Yuming Mao, Supeng Leng, Yejun He, and Yan Zhang. Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading. *IEEE Vehicular Technology Magazine*, 12(2):36–44, 2017.

[81] Toni Janevski. *Traffic analysis and design of wireless IP networks*. Artech House, 2003.

[82] Mascha Kurpicz, Anne-Cécile Orgerie, and Anita Sobe. How much does a VM cost? energy-proportional accounting in vm-based environments. In *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pages 651–658. IEEE, 2016.

[83] Leila Ismail and Huned Materwala. Escove: Energy-sla-aware edge–cloud computation offloading in vehicular networks. *Sensors*, 21(15):5233, 2021.

[84] Philipp Wiesner and Lauritz Thamsen. Leaf: Simulating large energy-aware fog computing environments. In *2021 IEEE 5th International Conference on Fog and Edge Computing (ICFEC)*, pages 29–36. IEEE, 2021.

[85] Young Choon Lee and Albert Y Zomaya. Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, 60(2):268–280, 2012.

[86] Franz Christian Heinrich, Tom Cornebize, Augustin Degomme, Arnaud Legrand, Alexandra Carpen-Amarie, Sascha Hunold, Anne-Cécile Orgerie, and Martin Quinson. Predicting the energy-consumption of mpi applications at scale using only a single node. In *2017 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 92–102. IEEE, 2017.

[87] Arun Vishwanath, Kerry Hinton, Robert WA Ayre, and Rodney S Tucker. Modeling energy consumption in high-capacity routers and switches. *IEEE Journal on Selected Areas in Communications*, 32(8):1524–1532, 2014.

[88] Vijay Sivaraman, Arun Vishwanath, Zhi Zhao, and Craig Russell. Profiling per-packet and per-byte energy consumption in the netfpga gigabit router. In *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 331–336. IEEE, 2011.

[89] Md Hussain, MM Beg, et al. Fog computing for internet of things (IoT)-aided smart grid architectures. *Big Data and cognitive computing*, 3(1):8, 2019.

[90] Christian Hinrichs, Sebastian Lehnhoff, and Michael Sonnenschein. Cohda: A combinatorial optimization heuristic for distributed agents. In *Agents and Artificial Intelligence: 5th International Conference, ICAART 2013, Barcelona, Spain, February 15-18, 2013. Revised Selected Papers 5*, pages 23–39. Springer, 2014.

[91] Christian Hinrichs and Michael Sonnenschein. A distributed combinatorial optimisation heuristic for the scheduling of energy resources represented by self-interested agents. *International Journal of Bio-Inspired Computation*, 10(2):69–78, 2017.

[92] Richard P. Brent. Efficient implementation of the first-fit strategy for dynamic storage allocation. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 11(3):388–403, 1989.

[93] Olena Skarlat, Matteo Nardelli, Stefan Schulte, Michael Borkowski, and Philipp Leitner. Optimized iot service placement in the fog. *Service Oriented Computing and Applications*, 11(4):427–443, 2017.

[94] Philipp Wintersberger, Anna-Katharina Frison, Andreas Riener, and Tamara von Sawitzky. Fostering user acceptance and trust in fully automated vehicles: Evaluating the potential of augmented reality. *PRESENCE: Virtual and Augmented Reality*, 27(1):46–62, 2019.

[95] Yiheng Feng, Chunhui Yu, Shaobing Xu, Henry X Liu, and Huei Peng. An augmented reality environment for connected and automated vehicle testing and evaluation. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1549–1554. IEEE, 2018.

[96] Arun Vishwanath, Fatemeh Jalali, Robert Ayre, Tansu Alpcan, Kerry Hinton, and Rodney S Tucker. Energy consumption of interactive cloud-based document processing applications. In *2013 IEEE International Conference on Communications (ICC)*, pages 4212–4216. IEEE, 2013.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

TABLE V: Comparison of costs and percentage differences

Cost	Percentage Difference with MERA (%)			
	Baseline (Optimized)	Greedy (Optimized)	Baseline (Default)	Greedy (Default)
Deadline	+14.38	+33.79	+16.56	+31.71
Processing	+5.51	-6.61	+2.97	-3.76
Storage	0.00	-0.03	+2.13	-2.56
RAM	-0.05	-0.11	-0.18	-2.38
Deployment	-36.79	-96.63	-48.98	-96.58
Communication	-75.61	+1.81	-90.13	-12.16
Energy	+1.85	+0.23	-0.14	-0.30
CO2	0.00	0.00	0.00	0.00

[97] Mikola Patlayenko, Olena Osharovska, and Valentyna Solodka. Comparison of lte coverage areas in three frequency bands. In *2021 IEEE 4th International Conference on Advanced Information and Communication Technologies (AICT)*, pages 212–215. IEEE, 2021.

[98] Ruben Mayer, Leon Graser, Harshit Gupta, Enrique Saurez, and Umakishore Ramachandran. Emufog: Extensible and scalable emulation of large-scale fog computing infrastructures. In *2017 IEEE Fog World Congress (FWC)*, pages 1–6. IEEE, 2017.

[99] Kuljeet Kaur, Tanya Dhand, Neeraj Kumar, and Sherali Zeadally. Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers. *IEEE wireless communications*, 24(3):48–56, 2017.

[100] Chuhao Qin and Evangelos Pournaras. Coordination of drones at scale: Decentralized energy-aware swarm intelligence for spatio-temporal sensing. *Transportation Research Part C: Emerging Technologies*, 157:104387, 2023.

APPENDIX

APPENDIX A: SUPPLEMENTARY MATERIAL

Figure. 11 illustrates the coefficient of variance and service provisioning cost in optimized routes scenario. In this scenario, all approaches exhibit reduced coefficients of variance compared to default routes, indicating that optimized routing strategies contribute to more consistent server utilization. This emphasizes the significance of efficient traffic route planning in minimizing variability and enhancing overall edge-to-cloud network performance. Particularly, the Baseline approach shows a more pronounced decrease in coefficients of variance, underscoring its effectiveness in achieving stable server utilization compared to the other approaches in the presence of optimized routes.

Table V illustrates the changes in service provisioning cost components for the Greedy and Baseline approaches compared to MERA, highlighting both improvements (indicated by +) and deteriorations (indicated by -). Table VI outlines the mathematical notations utilized throughout the paper.

TABLE VI: Mathematical notations

Notation	Meaning
C	Set of cloud nodes
F	Set of fog nodes
A	Set of services/vehicles
M	Set of access points in the communication network
s_i	Speed of vehicle i
l_{im}	Euclidean distance between vehicle i and access point m
P_{im}	Connection probability of vehicle i and access point m
t_{im}	Initial connection time of i with m
t^{w}_{im}	Waiting time of i to receive reply from m
τ	Interval between consecutive optimization problem-solving instances (seconds)
τ_{im}	Connection time of vehicle i with access point m (seconds)
δ	Service placement plan
x_{ij}	Binary placement decision for i on fog node j
x_{ic}	Binary placement decision for i on cloud node c
$d_{jj'}$	Propagation delay link (j, j')
$b^{u}_{jj'}$	Average uplink transmission rate of the link (j, j')
$b^{d}_{jj'}$	Average downlink transmission rate of the link (j, j')

continued on next column

Notation	Meaning
F^{p}_j	Processing capacity of node j (MIPS)
F^{m}_j	Memory capacity of node j
F^{s}_j	Storage capacity of node j
u_j	Processing power utilization of node j
μ_j	Service rate of one processing unit of node j (MIPS)
n_j	Number of processing units of node j
L^{p}_i	CPU demand of service i (million instruction per request)
L^{m}_i	Memory demand of service i (bytes)
L^{s}_i	Storage demand of service i (bytes)
η_i	Desired QoS level for service i
h_i	Delay threshold for service i
V^{m}_{ij}	Delay violation percentage for service i on node j connected to m
V_i	Delay violation percentage for service i on node j
e_{ij}	Delay of service i running on node j
w_{ij}	Waiting delay for i served by node j
q_i	Average size of requests of service i (bytes)
a_i	Average size of responses of service i (bytes)
z_{ij}	Arrival rate of service requests i to node j (request/second)
ζ^{p}_{ij}	Traffic arrival rate of service i to node j (instruction/second)
f_{ij}	Processing power ratio allocated to service i on node j
P^{n}_{δ}	Power consumption of networking equipment
P^{s}_{δ}	Power consumption of server machines
P^{i}_j	Power consumption of node j in idle mode (watt)
P^{a}_j	Active power consumption of node j (watt)
P^{i}_c	Power consumption of cloud node c in idle mode (watt)
P^{a}_c	Active power consumption of cloud node c (watt)
p^{r}_j	Ratio of renewable power supplied to fog node j (server or edge/core router)
p^{r}_c	Ratio of renewable power supplied to cloud node c (server or switch)
p^{r}_m	Ratio of renewable power supplied to access point m
θ_c	PUE of cloud center c
p^{f}_j	Power consumption of edge/core router j for data transfer (watt per byte)
p^{f}_m	Power consumption of access point m for data transfer (watt per byte)
P^{E}_{δ}	Non-renewable power consumption of fog-cloud infrastructure
R_c	Average carbon emission rate for electricity
C^{p}_j	Unit cost of processing at node j ($ per million instructions)
C^{s}_j	Unit cost of storage at node j ($ per byte per second)
C^{m}_j	Unit cost of RAM at node j ($ per byte per second)
$C^{\mathsf{c}}_{jj'}$	Communication cost of link (j, j') ($ per unit bandwidth per second)
C^{v}_i	Cost of deadline violation for service i ($ per request per %)
C^{r}	Unit cost of renewable power consumption supplied
C^{n}	Unit cost of non-renewable power consumption supplied
C^{f}	Unit cost of carbon footprint
O^{P}_{δ}	Cost of processing for plan δ
O^{S}_{δ}	Cost of storage for plan δ
O^{M}_{δ}	Cost of RAM usage for plan δ
O^{C}_{δ}	Cost of communication for plan δ
O^{D}_{δ}	Cost of deployment for plan δ
O^{V}_{δ}	Cost of deadline violation for plan δ
O^{E}_{δ}	Cost of energy consumption for plan δ
O^{F}_{δ}	Cost of carbon footprint for plan δ
L_{δ}	Local cost of service provisioning for plan δ
G	Global cost of service placement plans

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3552504

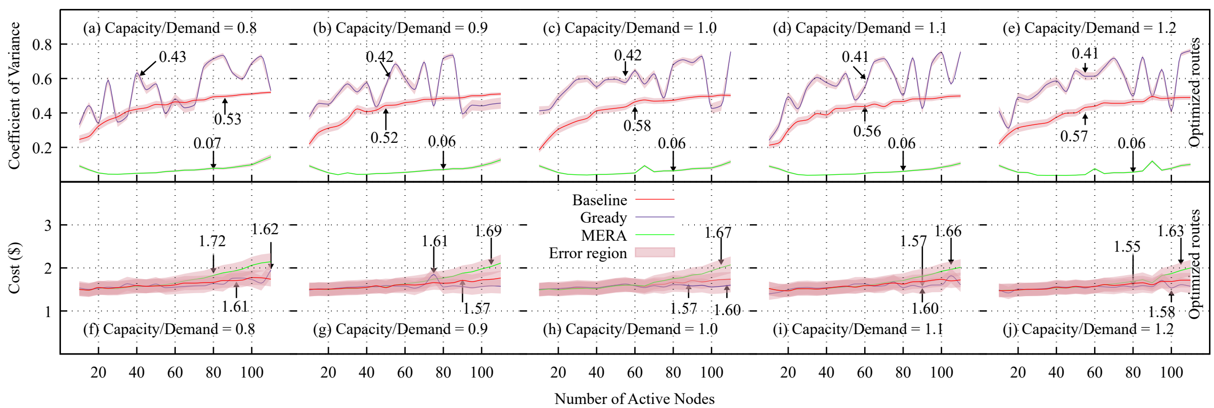

Fig. 11: The influence of edge-to-cloud capacity and size on load dispersion and service provisioning cost: The increase in capacity reduces costs and converges them across all approaches, the increase in the number of active nodes increases utilization variation in all methods.