

When ‘Computing follows Vehicles’: Decentralized Mobility-Aware Resource Allocation in the Edge-to-Cloud Continuum

Zeinab Nezami
School of Computing
University of Leeds
Leeds, United Kingdom
z.nezami@leeds.ac.uk

Emmanouil Chaniotakis
UCL Energy Institute
University College London
London, United Kingdom
m.chaniotakis@ucl.ac.uk

Evangelos Pournaras
School of Computing
University of Leeds
Leeds, United Kingdom
e.pournaras@leeds.ac.uk

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, 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 able to reduce utilization variance by more than 40 times, 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 emerged as the largest emitting sector of greenhouse gases, accounting for 27% of the UK’s total emissions¹, underscoring the need to prioritize CO2 emission reduction via traffic management and implementation of smart mobility services. With the advancements in Intelligent Transportation System (ITS), technologies such as connected and autonomous vehicles are envisioned to provide a safer, environmentally friendly, and convenient [1, 2] transportation ecosystem for the public. Vehicles are equipped with wireless communication capabilities to support a plethora of applications including rerouting, road safety, and location-based services [3, 4]. The imminent proliferation of Internet of Things (IoT) devices, including vehicles, are projected to surpass 41.6 billion by 2025 and generate up to 79.4 zettabytes of data, as estimated by the International Data Corporation. These developments raise concerns about the capacity of Information and Communication Technology (ICT) to respond

and adapt. Moreover, this upsurge in data generation coincides with the considerable energy consumption attributable to ICT, already surpassing 10% of global energy consumption and forecasted to exceed 20% by 2030 [5].

Besides the surge in the data generation, the mobility of vehicles stands as a pivotal characteristic within the IoT environment, introducing dynamism and uncertainty. This dynamic nature poses numerous challenges, particularly in the realm of ICT resource management for applications with stringent QoS requirements. Vehicle-to-Infrastructure (V2I) enhances data distribution effectiveness by shifting cloud services to network edges. Fog and mobile edge computing (MEC) [6, 7] present a novel paradigm for offloading computation from vehicles to local servers while facilitating location-aware collaborative data sharing [8] and traffic rerouting [9]. Moreover, the distributed architecture of fog computing (i.e., edge-to-cloud) reduces data traversal through the network, resulting in significant energy savings of 14% to over 80% compared to fully centralized cloud center architectures [10, 11].

Nevertheless, the distributed, heterogeneous, and resource-constrained nature of fog infrastructure poses challenges regarding its ability to host and manage (i.e., service placement) the dynamic and stochastic nature of vehicular networks. Concentrating smart mobility services at nodes near intersections or vehicle destinations to meet the demanding QoS requirements, such as Higher Definition Maps (HD Maps) and augmented reality applications, may result in node overloading, inefficient resource distribution, and potential overprovisioning for some services while others remain underprovisioned. In addition to workload balance and QoS parameters, different optimization aspects must be considered to address the service placement problem, given the presence of mobile users and their traffic dynamics [12, 13].

This paper pioneers a distributed service provisioning framework, aiming to dynamically harmonize for the first time the intricate interplay among QoS, service provisioning cost, and sustainability concerns within the edge-to-cloud ecosystem. The envisioned framework entails a reconfigurable edge-to-cloud computing infrastructure designed to balance

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

the load of smart mobility services in response to dynamic spatio-temporal mobility patterns. The key contributions of this paper encompass: (i) Development of a novel and practical open-source framework¹ for dynamic provisioning (deploying and migrating) of smart mobility services. (ii) Formulation of an optimization problem addressing the QoS and mobility-aware service placement challenge. (iii) Applicability of an efficient collective decision making algorithm to tackle the problem, and (iv) Extensive evaluation based on real-world edge-to-cloud settings and traffic traces from Munich to verify the cost-effectiveness and scalability of the framework.

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

II. RELATED WORK

The problem of service placement 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 [14, 15] and multi-objective [16, 17] optimization, linear programming [18, 19], Markov Decision Process (MDP) [17, 20], game theory [21, 22], and fuzzy logic [23], catering to diverse QoS criteria such as service delay [19, 22, 24–27], resource efficiency [25], energy consumption [26–30], economic cost [19, 26, 31], and system utility [14, 32]. These formulations lay the groundwork for developing service placement algorithms falling into categories such as exact [33, 34], heuristic [15, 35–37], meta-heuristic [24, 27, 30], and machine learning-based [31, 35, 38, 39] solutions. Static placement [18, 20, 23, 25, 30, 33, 34] 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 outlines most recent literature contributions, addressing the problem within a mobility-aware context, as summarized in Table I.

Zhan *et al.* [41] 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. They leverage Deep Reinforcement Learning (DRL) methods to optimize this model. An Integrated MEC framework is proposed by Awada *et al.* [40] for resource-aware offloading of 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.* [36] present a joint task offloading and resource

TABLE I: Literature review summary

Criteria/ Study	Optimization Criteria	Proposed Ap- proach	IoT Load	ICT Network	Traffic Network (mobility)
[31] [40]	SU SD, RU	DRL Bin-Packing	Synthetic Alibaba cluster trace	Synthetic -	- Synthetic
[36] [38] [27]	SD SD EC, SD	Heuristic DRL Greedy, Genetic algorithm	Synthetic - Synthetic	- Synthetic -	Synthetic Rome (GPS trajectory) San Francisco (real trajectory)
[41] [42] [22]	SD, EC SR CD, MC	DRL Approximation Approximation, Greedy	Synthetic Synthetic Synthetic	- Synthetic Synthetic	- Melbourne (real trajectory) 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.

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.* [22] 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.* [31] propose a vehicle-edge-cloud resource allocation framework aimed at maximizing system utility. This framework supports vehicle-to-vehicle offloading to enable vehicles with idle resources to engage in task deployment, and V2I offloading for load balancing. They employ a Multi-agent MDP-based DRL algorithm to make optimal scheduling decisions.

The current research [17, 22, 27, 31, 35, 36, 38, 40–42] on service placement typically focuses on a single or limited criteria. While 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, while most of the cited studies [27, 31, 36, 40] rely on central control plans or non-cooperative solutions [22], our approach stands out for its distributed cooperative nature, offering scalability, resilience, and flexibility.

III. SERVICE PLACEMENT FRAMEWORK

This paper introduces a dynamic service provisioning framework that integrates mobility and QoS considerations. It addresses the challenge of service provisioning by balancing factors such as load, QoS, cost, energy efficiency, and sustainability. The framework examines resource utilization

¹<https://github.com/DISC-Systems-Lab/Edge-Mobility-Cooptimization> (last accessed: April 2024).

across an edge-to-cloud continuum to meet the demanding requirements of smart mobility services, especially those with low-latency and high-bandwidth needs. 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 [18]. The section outlines the layered architecture of the framework, request handling, and practical considerations. Following this, the next section focuses on problem formulation, taking into account both fog nodes' individual and system-wide objectives.

A. Motivational Application

Based on the latest C-V2X roadmap of 5GAA¹, 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 Maps²) 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 this 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 forecasts a significant rise in Augmented Reality Head-Up Displays, with an estimated 15 million units shipped by 2025, including 11 million integrated into vehicles³. This paper focuses on passenger-facing HD Maps' augmented reality technology, resembling the 2020 Mercedes-Benz GLE display [43]. Continuous data flows from mobile devices to nodes hosting AR services incur significant communication and processing costs, demanding substantial computational resources and low-latency processing [44]. This poses challenges to the readiness of self-driving cars, as highlighted by Civil Maps co-founder Sravan Puttagunta. 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, conveying it as a continuous stream to higher-layer computing nodes for processing, and ending with command transmission to actuators [45].

B. Distributed Architecture

The system architecture for provisioning smart mobility services within V2I communication model is depicted in Figure 1. The bottom layer encompasses the traffic networks, 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 networks, the data layer provides realism to the framework, incorporating real-world communication settings for LTE access/edge networks, core networks, cloud data centers, and IoT workloads.

Above the data layer lies a layered ICT network architecture designed to host smart mobility services originating from the transportation sector. This architecture utilizes networking and computation nodes across the edge-to-cloud hierarchy, with vehicles positioned at the bottom layer and cloud centers at the top layer, complemented by distributed fog servers positioned between them to facilitate efficient data processing and communication. In the context of fog computing literature [46, 47], 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 routers at the edge of the core network. Core and edge routers, switches, and cellular LTE base stations (referred to as access points) serve as the connectivity infrastructure within and between layers. 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 [48]. Vehicles are equipped with a cellular radio interface [49] 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 framework 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 V2I infrastructure.

C. Handling Requests and practicality

Any vehicle requesting smart mobility services sends a service request, characterized by the parameters listed in Table V, together 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 vehicle's connecting access point) for processing. Every fog server is equipped with a software agent that is in charge of making decisions on the placement of the requests received by itself. The offloading decision may be determined by considering the information about the agent's neighborhood and exchanged information with neighboring servers. This process ensures that placement objectives are achieved while adhering to a

¹5G Automotive Association, available at <https://5gaa.org/about-us/> (accessed April 2024).

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

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

service level agreement (SLA). If the decision favors local execution, the request is processed by the node; otherwise, it is offloaded to the selected fog/cloud server. Upon acceptance, a container is created at the appointed server to process the request [50]. The mapping of services to the available servers (i.e., service placement) is updated periodically as vehicles move around. After completing the service computation, the host servers transmit the computing result to the nearest access point in the system, which then relays the response back to the vehicles.

The proposed framework is highly practical, operating without assumptions and requiring only minimal information about mobile nodes. For service placement, solely the incoming resource demand of mobile nodes to fog nodes is necessary. 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 the flexibility to either own its fog and edge resources or acquire them through rental arrangements with edge network providers such as AT&T, Nokia, Verizon, or other edge resource owners [51–53]. This work represents a rigor endeavor, grounded in real-world traffic and ICT architecture settings, specifically tailored for the vehicle-to-edge-to-cloud network paradigm. The authenticity of the data utilized in this study is ensured through selection from reputable sources, as detailed in Section VI.

IV. MOBILITY-AWARE SERVICE PLACEMENT PROBLEM FORMULATION

This section formally defines the mobility-aware service provisioning problem as an online task, presenting possible formulation of quadratic cost functions, acknowledging its NP-hard nature [25, 54]. We explore formulations for both (i) System-wide objectives, emphasizing workload balance, incentivization of renewable energy sources, and infrastructure energy efficiency, and (ii) Local objectives of service providers, focusing on minimizing service provisioning costs and meeting QoS requirements.

The solution to the service placement 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. \bar{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 V lists the notations used in the problem formulation.

A. System-wide Objectives

Due to the dynamic nature of vehicle mobility and the inherent unpredictability of the wireless medium, mobile

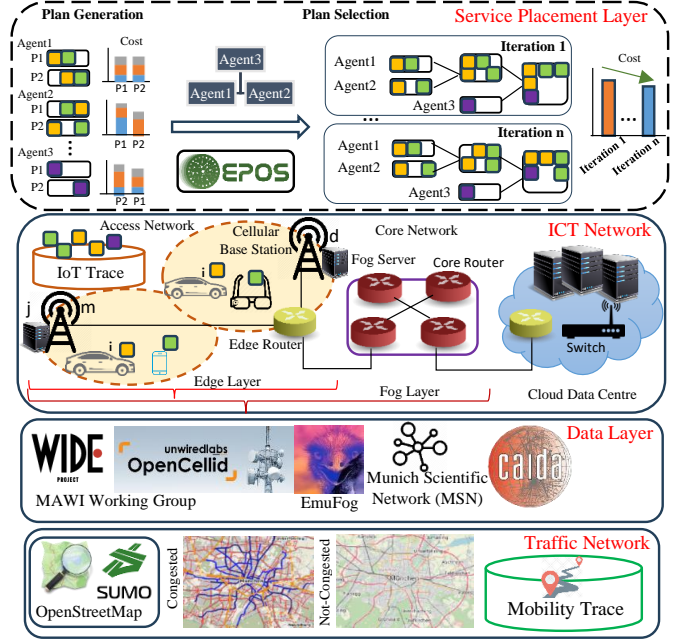


Fig. 1: Layered core components of mobility-aware service provisioning framework, consisting of: (1) Real-world traffic network, (2) Real-world data layer, (3) Vehicle-fog-cloud connectivity and computation infrastructure, and (4) Cooperative service placement orchestrating resource allocation.

IoT networks frequently experience fluctuations in traffic load [55]. 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 in a network is a prevalent strategy for network management and resource optimization, offering several advantages. It enhances energy efficiency by reducing operational costs and environmental impact, optimizes resource usage, enhances fault tolerance, and supports network scalability. However, this approach poses challenges such as dynamic application placement, load balancing, and temporary network disruptions. Careful planning and monitoring are essential for successful implementation. This research proposes a load-balancing strategy involving selectively deactivating network nodes and redistributing the workload among the remaining active nodes. This research concern is not only workload balance and reducing energy consumption but also promoting renewable energy with incentives. According to International Energy Agency (IEA), generating capacity from renewable energy sources of the world will reach 8500 GW by 2050. Energy harvesting from renewable resources such as wind and solar power is promising for increasing the efficiency and lifetime of edge-to-cloud network and making network more sustainable from the environmental point of view [56].

1) *Workload Balance:* Leveraging fog nodes in dynamic transport environments can enhance resource efficiency within edge-to-cloud networks and facilitate the execution of time-sensitive services. However, the individual objectives of fog nodes 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 network 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, nodes remain highly available to efficiently handle incoming requests, thereby reducing the likelihood of delayed responses to critical situations [57–59]. 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 distribution balance across all nodes, measured by the variance in terms of CPU and RAM usage:

$$\text{Min} \sqrt{\frac{1}{|F| + |C|} \sum_{j=1}^{|C \cup F|} \left(\frac{\zeta_j}{F_j^p} - \frac{\bar{\zeta}_j}{\bar{F}_j^p} \right)^2} \quad (1)$$

2) *Energy Consumption versus Workload Balance*: The framework empowers dynamic activation/deactivation of fog servers based on service demand, along with adjusting the participation level of various nodes in service hosting. To achieve this, an *Infrastructure Planner* is defined, integrating Traffic Monitoring Agents to observe incoming traffic rates (e.g. see earlier work [60]). 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 [61, 62]. 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.

3) *Renewable Energy Incentives*: According to Li et al. [63], renewable energy can fully support the reliable operation of edge devices using a microgrid (solar-wind hybrid energy system). 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 [64] 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) between network nodes' utilization and a target incentive vector based on the ratio of energy supplied from renewable sources (MIN-RMSE).

$$\text{Min} \sqrt{\frac{\sum_{j=1}^{|F \cup C|} (u_j - p_j^r)^2}{|C| + |F|}} \quad (2)$$

B. Local Objective

The deployment of an application across the fog-cloud network can significantly influence its non-functional aspects such as operational costs and performance [65]. The overall cost of executing a typical service placement plan δ in fog-cloud infrastructure encompasses the following components: processing cost ($O^P\delta$), RAM usage cost ($O^M\delta$), storage usage cost ($O^S\delta$), container deployment cost ($O^D\delta$), communication cost ($O^C\delta$), energy consumption cost ($O^E\delta$), and carbon footprint cost ($O^V\delta$). As formulated in Equation (3), this work combines these parameters with a penalty cost for service delay ($O^V\delta$), serving as a QoS measurement. Other expenses can also be identified such as security safeguards which are not the focus of this paper.

$$L_\delta = O_\delta^P + O_\delta^M + O_\delta^S + O_\delta^D + O_\delta^C + O_\delta^E + O_\delta^F + O_\delta^V \quad (3)$$

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 resource placement plan δ .

$$O_\delta^P = \sum_{i=1}^{|A|} \sum_{j=1}^{|F \cup C|} x_{ij} \cdot z_{ij} \cdot L_i^P \cdot C_j^P \cdot \tau \quad (4)$$

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.

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

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

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 [66], 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 .

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

The delay of the service i includes three components as *propagation delay*, *transmission delay*, and *waiting time* (processing delay plus queuing delay) [18, 67]. 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's host node j over time. These changes occur due to connection handoffs and handovers [68]. Consequently, this dynamic nature of connections results in different 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 vehicle's path 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:

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

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 [69]. It is calculated as follows:

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

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) [70] 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) [18, 71].

$$w_{ij} = \frac{P_{ij}^Q}{F_j^p \cdot f_{ij} - \zeta_{ij}} + \frac{1}{\mu_j \cdot f_{ij}}, \quad (10)$$

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

$$P_{ij}^Q = \frac{(n_j \cdot \rho_{ij})^{n_j}}{n_j!} \cdot \frac{P_{ij}^0}{1 - \rho_{ij}}, \quad (11)$$

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

$$P_{ij}^0 = \left[\sum_{c=0}^{n_j-1} \frac{(n_j \cdot \rho_{ij})^c}{c!} + \frac{(n_j \cdot \rho_{ij})^{n_j}}{n_j!} \left(\frac{1}{1 - \rho_{ij}} \right) \right]^{-1} \quad (12)$$

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 service¹ i is denoted by $\eta_i \in (0, 1)$ 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 :

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

3) *Cost of service deployment*: Given the unpredictable 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 between performance and cost, this paper 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.

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

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

4) *Cost of Communication*: This cost encompasses the overhead associated with uploading and downloading data, which includes the transmission of requests and responses. To quantify this cost, two factors must be taken into consideration: (i) It is possible that a service is hosted on a node different from the node it is directly connected to. Consequently, the data required for processing needs to be transferred from the connecting access point to the host node, (ii) Due to mobility, a vehicle may traverse multiple access points during service

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

offloading/processing. Consequently, the links and networking devices connecting this vehicle to its service's host may change along its route. These result in varying communication costs during a particular journey over a specific time period. 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 node 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 may communicate with each other, and the service is relayed through the backhaul network [72, 73]. The frequent movement of vehicles often results in intermittent connectivity, posing 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.

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

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:* According to Leslie Hook and Dave Lee of the Financial Times “the combined annual electricity consumption of the major tech companies Amazon, Google, Microsoft, Facebook, and Apple equals that of the entire country of New Zealand”. A significant share of this consumption is attributed to the energy-intensive operations particularly in handling computationally intensive requests and will grow as the rise of artificial intelligence demands more computing power [74]. The total energy consumption in an IoT network comprises static and dynamic components. The static part represents energy usage when resources are idle [75], while the dynamic component is determined by the current workload on active virtual machines and containers [10, 12]. As the static part is not affected by a service placement policy we ignore it in our model and explore the cost of the dynamic part. The dynamic energy cost stems from two main sources of networking and servers energy consumption, which are further illustrated in the following.

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

Dynamic power consumption of physical machines depends mainly on CPU utilization and can be modeled as a linear function [45, 76–79] (see Equation (17)). Our fog servers are in the form of nano data centers that consist of one single physical machine without any extra additional ICT equipment (i.e., fans and links) [10]. However data centers have non-ICT equipment such as cooling system. To account for the energy consumption of non-ICT equipment, the Power Usage Effectiveness (PUE), the ratio between the total facility and

IT equipment power consumption, is utilized as a widely recognized indicator for data center energy efficiency.

$$P_{\delta}^s \triangleq \sum_{j=1}^{|F|} (P_j^a - P_j^i) \cdot u_j \cdot [(1 - p_j^f) \cdot C^n + p_j^f \cdot C^r] + \sum_{c=1}^{|C|} \theta_c \cdot (P_c^a - P_c^i) \cdot u_c \cdot [(1 - p_c^r) \cdot C^n + p_c^r \cdot C^r], \quad (17)$$

where the parameter $(P_j^a - P_j^i)/F_j^P$ determines the incremental power consumption per unit load on the servers and $u_j = \frac{z_{ij}}{F_j^P}$.

To compute the dynamic energy consumption of the telecommunication network linking physical servers with vehicles, as depicted in Figure. 1, we consider three main types of networking devices: edge routers, core routers, and switches located within data centers. When a networking equipment carries traffic, it consumes load-dependent power for packet processing and also for storing and forwarding the payload [80, 81]. Consequently, if the router j (the router connecting the fog server j to the network) receives z_{ij} requests for service i during the time period τ , it processes $z_{ij} \cdot (a_i + q_i)$ bytes in such interval. Note that the 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 fog-cloud infrastructure as a result of the placement plan δ .

$$P_{\delta}^n = \sum_{i=1}^{|A|} \sum_{j=1}^{|F|} x_{ij} \cdot z_{ij} \cdot p_j^f \cdot (a_i + q_i) \cdot [(1 - p_j^f) \cdot C^n + p_j^f \cdot C^r] + \sum_{i=1}^{|A|} \sum_{c=1}^{|C|} \theta_c \cdot x_{ic} \cdot z_{ic} \cdot p_c^f \cdot (a_i + q_i) \cdot [(1 - p_c^r) \cdot C^n + p_c^r \cdot C^r] + \sum_{i=1}^{|A|} \sum_{j=1}^{|F \cup C|} \sum_{m=1}^{|M|} x_{ij} \cdot P_{im} \cdot z_{ij} \cdot p_m^f \cdot (a_i + q_i) \cdot [(1 - p_m^r) \cdot C^n + p_m^r \cdot C^r] \quad (18)$$

In Equation (18), the first segment 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.

6) *Cost of carbon footprint:* According to the Shift Project report¹, carbon emission from information technology infrastructure and data servers supporting cloud computing now surpasses those from pre-Covid air travel. The carbon footprint

¹Environmental 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)

is a critical consideration in server equipment deployment, and this study incorporates its cost into the optimization process, as shown in Equation (19). The calculation only accounts for power consumption from non-renewable sources (P_{δ}^E), following the U.S. Energy Information Administration's classification of clean energy sources as carbon neutral.

$$O_{\delta}^F = C^f \cdot R_c \cdot P_{\delta}^E \cdot \tau, \quad (19)$$

where R_c is the average carbon emission rate for electricity (kilogram per kilowatt-hour) [82].

C. Constraints

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

$$\sum_{i=1}^{|A|} L_i^s \cdot x_{ij} < F_j^s, \text{ and } \sum_{i=1}^{|A|} L_i^m \cdot x_{ij} < F_j^m, \forall j \in F \quad (20)$$

$$\sum_{i=1}^{|A|} L_i^s \cdot x_{ic} < F_c^s, \text{ and } \sum_{i=1}^{|A|} L_i^m \cdot x_{ic} < F_c^m, \forall c \in C \quad (21)$$

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

$$\zeta_{ij} < F_j^p \cdot f_{ij}, \text{ and } \zeta_{ic} < F_c^p \cdot f_{ic} \forall j \in F, \forall c \in C, \forall i \in A \quad (22)$$

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

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

D. Final Optimization Formulation

To address the overall problem, all eight local cost functions are considered initially; however, in certain scenarios, some costs can be omitted if needed, while others may play a dominant role in the overall cost summation. Note that in this optimization problem, we only consider 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 the optimization problem, since this is usually outside the control of service providers. Achieving both system-wide and local objectives requires coordination among agents and cannot be achieved solely through local optimization. Agents must select placement plans that meet multiple complex criteria, including minimizing quadratic cost functions. The next section introduces a framework that autonomously manages these diverse objectives, leveraging agents' local parameters to effectively navigate trade-offs.

V. COOPERATIVE SERVICE PLACEMENT ALGORITHM

This section studies how collaboration among agents can enhance system-wide and individual benefits. It outlines a distributed service provisioning algorithm, addressing the challenges of the problem in periodic intervals, avoiding centralized servers' bottlenecks, and implementing a strategy for machine activation and deactivation to ensure scalability against

bursty resource usage patterns. This work leverages EPOS (Economic Planning and Optimized Selections) [83, 84], a decentralized multi-agent system designed to tackle complex multi-objective discrete-choice combinatorial problems via a collective learning approach. As shown in Figure. 1, fog agents collaborate to select one service placement plan for their received requests in the two steps of plan generation and plan selection, aiming to meet both local (minimization of service provisioning cost) and global (minimization of utilization variance) objectives.

Initially, each fog agent autonomously generates multiple mappings (i.e., possible service placement plans with different costs). In this step, agents prioritize minimizing the service placement cost of their own received requests to enhance QoS and keep the service provisioning cost low. The possible plans encode selected hosts using a binary vector and resource utilization with a vector of real values [25]. 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 objectives (MIN-COST) and global objectives (MIN-VAR). Agents autonomously organize into a tree overlay topology to structure their interactions and facilitate information exchange [83, 84], enabling cooperative optimization. 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. In each iteration, agents refine their chosen plans by combining the two 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.

$$\lambda \cdot L^t + (1 - \lambda) \cdot G^t, \quad (24)$$

where $\lambda \in [0, 1]$. 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 at the iteration $t - 1$, which is the sum of all utilization plans of the agents in the network. The global cost functions (MIN-VAR) and the local cost function (MIN-COST) are formulated as follows:

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

where $G^t, L^t \in \mathbb{R}$ and $l(\cdot)$ 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.

A. Computational Efficiency and Time Complexity

The time complexity of the proposed algorithms described in Section V are discussed below. The Greedy algorithm employs a local plan selection approach. The sorting of fog nodes by agents results in a computational complexity of $O(|F| \cdot \log |F|)$ for each service. This step runs at most $|A|$ times for all services. Consequently, the overall complexity of Greedy for all services is $O(|A| \cdot |F|^2)$. The computational complexity of EPOS is contingent upon the number of iterations performed. The computational complexity along the critical path relies on the network size. In the case of EPOS, it's the tree height that influences the computational complexity, which is logarithmic to the number of agents, $\log |F|$. For each agent which is $O(|\delta| \cdot t)$. Consequently, the overall complexity for all services is $O(|\delta| \cdot t \cdot \log |F|)$. In the next section, we evaluate the algorithm experimentally using real-world data.

VI. EVALUATION

The paper assesses the proposed framework through the examination of six scenarios. Two types of mobility profiles, encompassing vehicles' default and optimized routes, are subjected to three service placement approaches including EPOS, Greedy, and Baseline. In the baseline approach, each fog server evaluates its own resource availability, opting for local deployment if feasible; otherwise, requests are directed to the cloud. The Greedy algorithm, akin to the widely-used First Fit approach in edge and cloud computing service placement [85, 86], aims to efficiently allocate resources while balancing optimization with computational efficiency. Upon receiving a request, a fog server assesses available resources nearby with adequate capacity for the service, computes the execution cost 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.

Our experiments are conducted within Munich, Germany. An area of 7000 m^2 of the city center is selected as the test area, shown in Figure. 2a. The traffic distribution pattern is derived from a real-world traffic trace and Sumo traffic simulator¹, employing two routing algorithms: (i) the Dijkstra algorithm, which determines the shortest path for vehicles in terms of travel time and distance (default routes), and (ii) the dualroute algorithm², which iteratively generates and refines vehicle routes based on observed traffic conditions to enhance traffic flow and alleviate congestion (optimized routes). Figure. 3 shows the number of vehicles that fall into the area and connect to the edge-to-cloud servers using the mentioned two algorithms during a three hour time span.

A. Simulation setting

The incoming workload to the experiments originates from two sources (i) IoT service requests, indicated in Figure. 4,

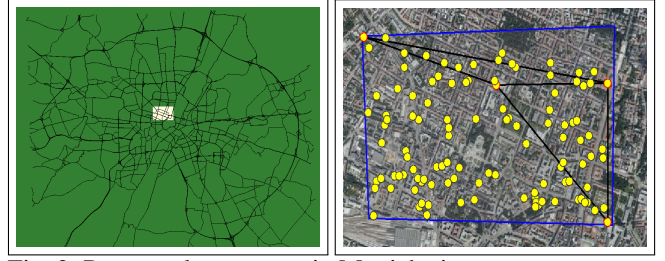


Fig. 2: Rectangular test area in Munich city center, core routers connected with black lines and LTE access points within the test area are highlighted

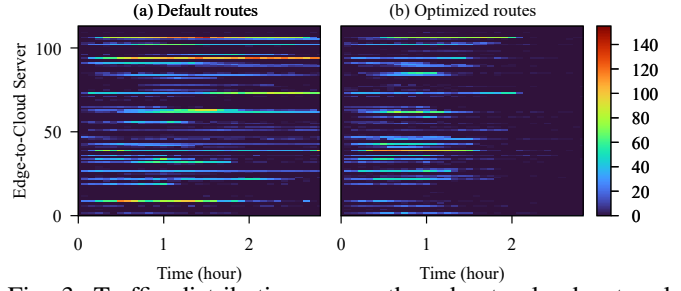


Fig. 3: Traffic distribution across the edge-to-cloud network over a three-hour period: the number of connected vehicles to each access point, along with their presence time, is higher in default routes compared to optimized routes.

referred to as IoT profiles and (ii) Vehicles' traffic, indicated in Figure. 3, referred to as mobility profiles. The IoT service requests come from the real-world traffic traces of Measurement and Analysis on the WIDE Internet (MAWI) traffic repository³. The MAWI archive, derived from the WIDE Internet backbone connecting Japanese academic and research institutions to the Internet, serves as an appropriate dataset for modeling incoming IoT traffic rates to fog nodes [18]. Capturing daily traces at the transit link of WIDE to its upstream ISP, the archive includes both TCP and UDP 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 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 [87, 88]. The IoT workload is adjusted every 15 minutes based on the IoT trace data, each 15-minute profile is divided into three 5-minute runs. Furthermore, the number of vehicles present in the test area is updated every 5 minutes. At the onset of each 5-minute interval, the placement algorithms are executed, and it is assumed that the resulting placements remain constant throughout the duration of the interval.

1) *Edge-to-cloud network*: The experiments are conducted on the world's largest Open Database of Cell Towers OpenCel-

¹Sumo, available at <https://sumo.dlr.de/docs/index.html> (accessed April 2024).

²Duarouter, available at <https://sumo.dlr.de/docs/duarouter.html> (accessed April 2024).

³MAWI Working Group Traffic Archive, available at <http://mawi.wide.ad.jp/mawi> (accessed April 2024).

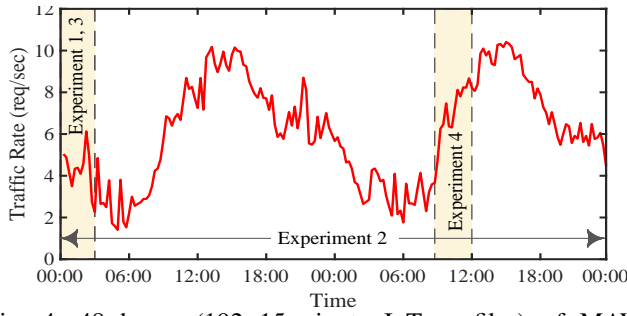


Fig. 4: 48 hours (192 15-minute IoT profiles) of MAWI trace data (April 12-13, 2017) serve as the input workload, delineating the experimental time intervals

IID¹ 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. Considering the high population density urban core of Munich downtown, we assume every cell site connects the adjacent vehicles to the network with the coverage range of 700 meters [89] in the frequency band 2.6 GHz. The network features one cloud center within a 10-hop distance from access points with a propagation delay $U(15, 35)$ ms [18], requires one Cisco Ethernet switch [10]. The number and location of core routers within the test area is determined using the Macroscopic Internet Topology Caida dataset², EmuFog emulator [90], and the Munich Scientific Network (MWN)³. Given the partially meshed topology of the Internet backbone, balancing cost-effectiveness with redundancy and efficiency, we depict four core routers strategically connected within the area, as shown in Figure 2b.

TABLE II: Experimentation parameters [18, 91]

Parameter	Value	Parameter	Value
Test Area	$2101.98 * 3313.97 m^2$	L^m	$U(2-400)$ MB
$ C $	1	L^b	$U(50,200)$ MI per req
$ F $	114	L^s	$U(50-500)$ MB
$ S $	1	h_i	10 ms
$ M $	109	q_i	$U(10-26)$ KB
$ A $	9788,10112	a_i	$U(10-20)$ Byte
$ N $	109	$ \delta $	20
s_i	$\leq 120 km/h$	θ_c	1.2^4
τ	300 seconds	p_c^r	0.85^5
λ	$\{0.05n \mid n \in \mathbb{N}_0, 0 \leq 0.05n \leq 1.0\}$	p_j^r	Beta(2, 0.5)
C_j^p	edge 0.6, otherwise 0.2 USD per 1M req ⁶	R_c	380 g CO ₂ eq/kWh ⁷
C_j^m	$21 * 10^{-10} / 128$ USD per MB per ms ⁸	$b_{jj'}^d$	[LTE: 0.072, Edge/Core: 10/100, Cloud: 100] Gbps
C_j^s	$U(0.021,0.023)$ USD per GB per month ⁹	$b_{jj'}^u$	[LTE: 0.012, Edge/Core: 1/10, Cloud: 100] Gbps
$C_{jj'}^c$	$U(0.01,0.03)$ 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) ¹⁰	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 ¹¹	C_c	17.27 € per tonnes ¹²
C^n	0.905 USD per kWh ¹³	C^r	294 € per MWh ¹⁴

¹OpenCellID, available at <https://opencellid.org> (accessed April 2024).

²Macroscopic Internet Topology Caida dataset, available at <https://www.caida.org/data/internet-topology-data-kit/release-2019-04.xml> (accessed April 2024).

³The Munich Scientific Network, available at <https://www.lrz.de/services/netz/> (accessed April 2024).

⁴Cloud Computing, Server Utilization, & the Environment, available at <https://aws.amazon.com/blogs/aws/cloud-computing-server-utilization-the-environment/> (accessed April 2024).

⁵International Energy Agency, available at <https://www.iea.org/energy-system/buildings/data-centres-and-data-transmission-networks> (accessed April 2024).

⁶AWS, available at <https://aws.amazon.com/lambda/pricing/> (accessed April 2024).

Tables IV and II outline the specifications of the networking devices and the traffic capacity of their connecting links respectively. The machines detailed in Table III exhibit comparable characteristics to the c5.metal instances introduced by Amazon in 2019¹⁵, serving as our fog and cloud servers. Furthermore, it is assumed that the storage/processing capacity of the cloud center is limitless. In 2020, renewable energy accounted for 50.9% of Germany's electricity generation¹⁶. Additionally, Munich, known for its focus on solar energy, aims to transition to a fully renewable energy system by 2025¹⁷. For our analysis, we assume that all other networking and computing resources in the network are supplied by the same power utility, thus 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 QoS. Nevertheless, deploying containers typically take less than 50 ms [92], 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) Service Credits, which are determined as a percentage of monthly bill for Amazon EC2 instances. The instance chosen for analysis is t2.nano¹⁸, featuring 1 vCPU, 0.5 GB of memory, and an On-Demand hourly credit of 0.0058 USD, aligning closely with the characteristics of our 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

⁷Current emission in Germany, available at <https://www.nowtricity.com/country/germany/> (accessed April 2024).

⁸AWS Lambda Pricing for a Serverless Application, available at <https://www.clickittec.com/devops/aws-lambda-pricing/> (accessed April 2024).

⁹AWS, available at <https://aws.amazon.com/s3/pricing/> (accessed April 2024).

¹⁰The 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).

¹¹AWS SLA, available at <https://aws.amazon.com/compute/sla/> (accessed April 2024).

¹²Carbon Taxes in Europe, available at <https://taxfoundation.org/carbon-taxes-in-europe-2022/> (accessed April 2024).

¹³Germany electricity prices, available at https://www.globalpetrolprices.com/Germany/electricity_prices/ (accessed April 2024).

¹⁴S&P Global Commodity Insights, available at <https://www.spglobal.com/commodityinsights/en/market-insights/latest-news> (accessed April 2024).

¹⁵Amazon EC2 C5 instances, available at <https://aws.amazon.com/blogs/aws/now-available-new-c5-instance-sizes-and-bare-metal-instances/> (accessed April 2024).

¹⁶Wikipedia, available at https://en.wikipedia.org/wiki/Renewable_energy_in_Germany Statistics (accessed April 2024).

¹⁷Reuters, available at <https://www.reuters.com/business/sustainable-business/germany-aims-get-100-energy-renewable-sources-by-2035-2022-02-28/> (accessed April 2024).

¹⁸AWS, available at <https://aws.amazon.com/ec2/pricing/on-demand/> (accessed April 2024).

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 [80, 81, 91]

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

the realms of demand variability and resource limitations, respectively. The fourth experiment scrutinizes the correlation between costs and incentivization strategies toward renewable energy sources. 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%.

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 correlation 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.

Figures. 5a, b and 7 depict the edge-to-cloud nodes {CPU|memory} utilization variance and CPU utilization respectively, illustrating the load distribution across the network for the three tested approaches. Greedy demonstrates the highest variance by concentrating the load on specific nodes. In contrast, the Baseline, which utilizes all nodes receiving requests, achieves a more distributed workload with lower variance. EPOS, by spreading the workload evenly across a broader range of network nodes, offers the most effective balance. It is evident that in the default routes scenario, both the baseline and greedy approaches exhibit considerable disparities compared to EPOS. The baseline approach, on average, shows a 40-fold increase in variance, while the greedy approach displays a much higher difference of approximately 240 times. In the optimized routes scenario, while the baseline's increase remains notable (approximately 28 times higher), the greedy approach demonstrates even larger variance (about 373 times higher) compared to EPOS. This observation underscores the effective workload distribution of EPOS, contrasting with the distribution inefficiencies of the baseline and greedy approaches. In the optimized scenario, the variance of EPOS is only half that of the default scenario, confirming its excellence, particularly in uncongested networks.

Figure. 5c, d depicts the cost of service provisioning across

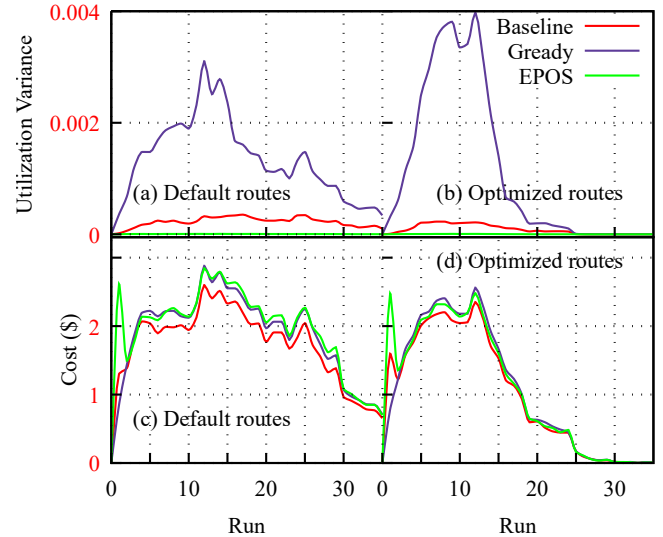


Fig. 5: The influence of traffic patterns on optimization objectives reveals distinct differences in workload distribution among placement approaches, while service provisioning costs converge.

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 EPOS, while Greedy shows a slight advantage (3%) over EPOS. However, in optimized route scenarios, these differences diminish, with Baseline demonstrating 6% and Greedy 1% lower cost compared to EPOS. Moreover, in the optimized scenario, the costs decrease by 45%, 42%, and 44% for EPOS, Baseline, and Greedy, respectively. This emphasizes the significance of uncongested traffic networks in reducing costs within ICT networks and enabling more efficient resource optimization. Across both optimized and default routes scenarios, the differences in storage, memory, energy, and CO2 costs between Baseline and Greedy approaches compared to EPOS are lower than 2.4%, with negligible variations observed. Initially, EPOS 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 EPOS. Conversely, the Greedy approach,

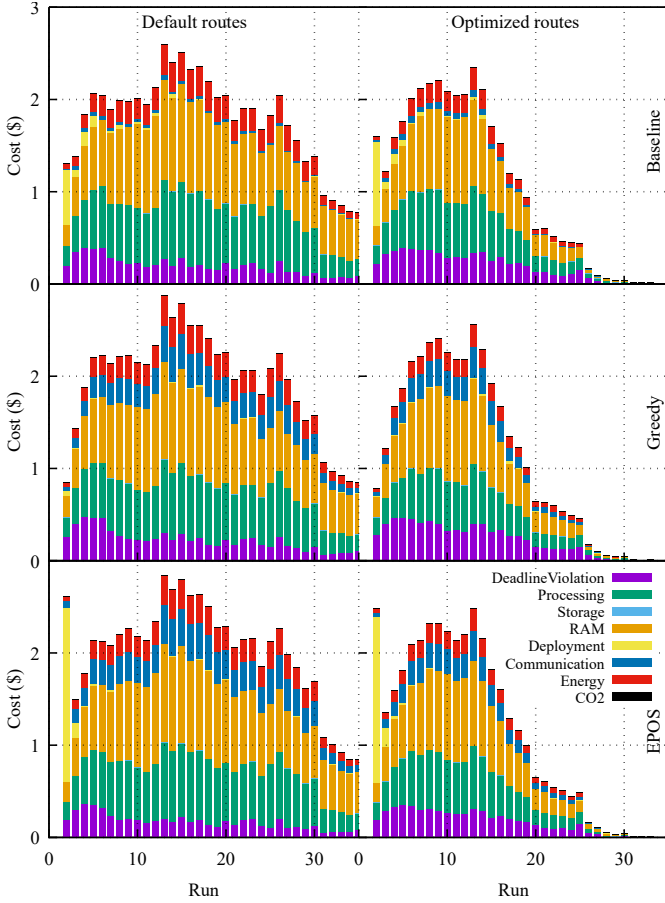


Fig. 6: EPOS outperforms other approaches in deadline violation, baseline excels in communication cost, and greedy offers the lowest deployment cost, with other costs being nearly similar.

which deploys services on a few specific nodes, exhibits a comparatively lower cost, with a decrease of 97% compared to EPOS. The cost of deadline violation is lower in EPOS, as it balances service distribution across the network, ensuring adequate processing power for each service. In contrast, Greedy's utilization of only a few nodes leads to higher queuing and execution delays for services. Additionally, EPOS's distribution of services closer to the sources mitigates communication delays. The baseline approach incurs a deadline cost approximately 14% to 17% higher than EPOS, while the greedy approach exhibits a more substantial increase, ranging from 31% to 34%. This underscores EPOS's effectiveness, particularly in time-sensitive applications such as augmented reality. The offloading of services across the network results in higher communication cost for both EPOS and Greedy compared to Baseline. The EPOS 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 VI in Appendix A compares the service provisioning costs of the baseline and greedy approaches to EPOS.).

The Greedy approach highlights the tendency of agents to prioritize their individual objective without coordinating to-

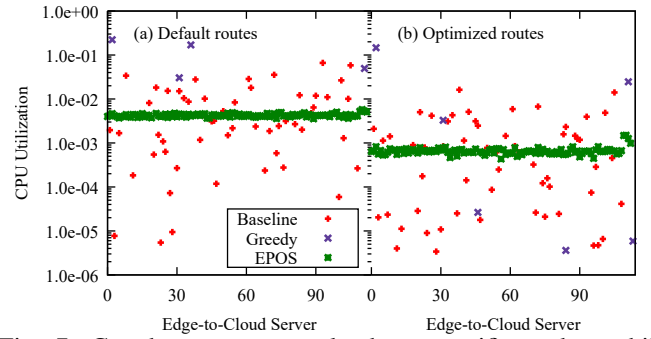


Fig. 7: Greedy concentrates load on specific nodes, while Baseline distributes workload across all receivers. EPOS achieves the most balanced workload distribution.

wards the system-wide goal, leading to striking fragmentation over time as certain nodes handle a significant workload, while others remain underutilized. Baseline, although it provides the least placement cost, results in a better load distribution yet unbalanced. EPOS, on the other hand, provides an even balance of workload at a slightly higher execution cost than baseline. However, this increased cost can be effectively managed by storing frequently used services locally by service providers.

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 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 costs. Conversely, with increasing load, costs diverge, especially noticeable in default route scenarios imposing heavier loads on the ICT network. At the peak IoT load (window 154), the Greedy approach is approximately 34% more costly than the baseline, decreasing to about 15% in optimized routes. Comparing EPOS with the baseline reveals a 10% increase in the optimized case and a 15% increase in the default scenario. Consequently, EPOS demonstrates greater scalability and robustness to IoT traffic peaks compared to Greedy in terms of service provisioning cost.

In terms of utilization variance, EPOS, Baseline, and Greedy experience reductions of approximately 46%, 60%, and 51%, respectively, when transitioning from default to optimized routes. This highlights a significant decrease in variability achieved with optimized traffic configurations. Regarding service provisioning cost, in the default routes scenario, EPOS, 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.

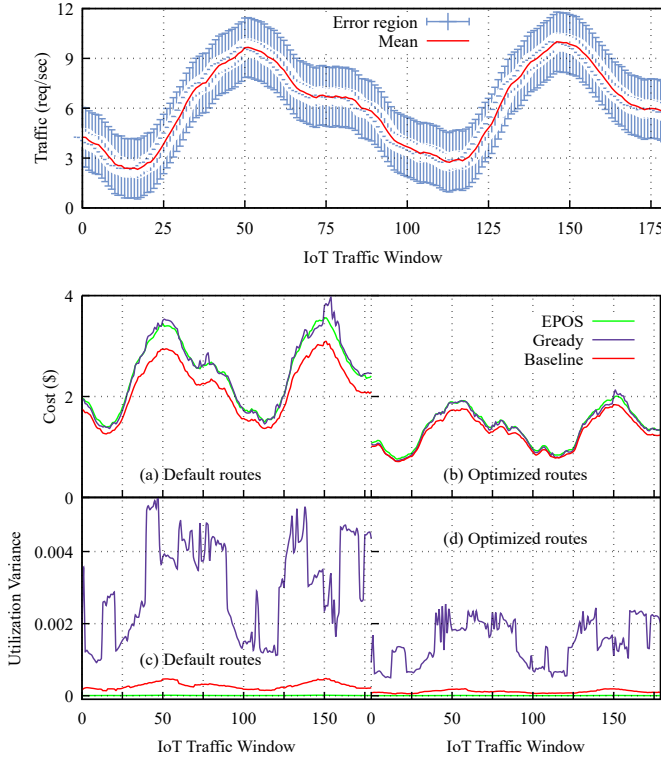


Fig. 8: Traffic variation across 12-profile IoT traffic windows influences load balance and service provisioning cost. Approaches diverge notably during traffic peaks, with Greedy exhibiting the largest fluctuations

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 representing the average outcome derived from these 10 samples of network nodes for default routes scenario.

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, resulting in variable server utilization. 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

EPOS 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 significantly reduces service placement costs for all approaches, benefiting both service providers and users, as well as the ICT network. While the Greedy and Baseline approaches exhibit a consistent relationship across variations in the number of nodes and capacity-to-demand ratio, EPOS experiences a gradual cost increase 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 compared to Greedy and Baseline can reach up to 25%. 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 (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 and deployment expenses.

4) *Experiment 4*: In this experiment, our proposed EPOS approach prioritizes renewable energy sources over non-renewable powered servers, as showcased by the logarithmic Figure. 10a, b. EPOS evaluates the effectiveness of this prioritization by measuring RMSE between the utilization and renewable power ratio of each server across the entire network. In the logscale 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 according to their utilization. The logarithmic scale enhances the visualization of the relationship between these variables, especially considering the wide range of values with different scales. The upward trend observed in the EPOS 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 minimizing service provisioning costs, respectively, which lead to deviations from the target signal in these approaches.

Figure. 10c, d illustrates the service provisioning costs associated with various approaches. Given the upward trend in incoming IoT traffic, as depicted in Figure. 4, these costs are notably higher compared to experiment 1 (with almost stable IoT traffic). Specifically, in the default scenarios, we observe an increase of 29%, 27%, and 39% for EPOS, Baseline, and Greedy, respectively. In contrast, in the optimized scenario, the increases are relatively lower, at 22%, 19%, and 19% for EPOS, Baseline, and Greedy, respectively. Furthermore, EPOS maintained consistent average cost differences compared to Baseline and Greedy, with Experiment 1 showing a 12% difference with Baseline and 4% with Greedy.

VII. CONCLUSION

The 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

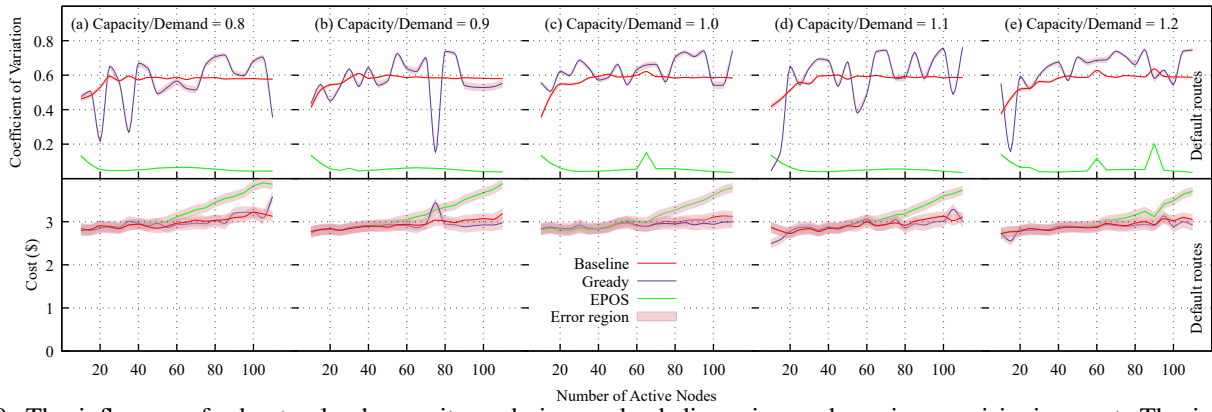


Fig. 9: The influence of edge-to-cloud capacity and size on load dispersion and service provisioning cost: The increase in capacity reduces costs and converges them across all approaches, the increase in the number of active nodes decreases utilization variation in EPOS while increasing it in other methods.

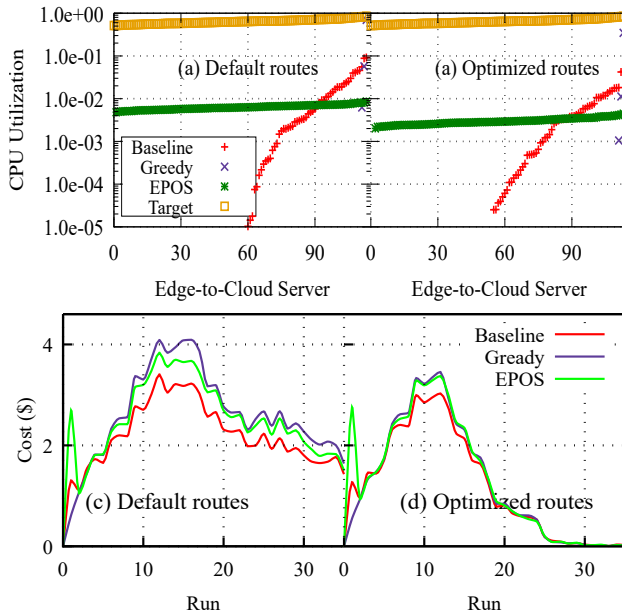


Fig. 10: EPOS loads nodes in accordance with their renewable energy penetration rate, while baseline deploys services locally, sacrificing load alignment for cost reduction. Greedy, on the other hand, fails to achieve superiority in either objective.

and 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 V2I ecosystem. By optimizing QoS, service provisioning costs, workload balance, and sustainability considerations, this framework offers a novel approach to navigating 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 enable self-adaptive ICT

infrastructures.

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] Nicola Jones. How to stop data centres from gobbling up the world’s electricity. *Nature*, 561(7722):163–167, 2018.
- [6] 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.
- [7] 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.
- [8] Dimas Satria, Daihee Park, and Minho Jo. Recovery for overloaded mobile edge computing. *Future Generation Computer Systems*, 70:138–147, 2017.
- [9] 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.

- [10] Ehsan Ahvar, Anne-Cécile Orgerie, and Adrien Lebre. Estimating energy consumption of cloud, fog and edge computing infrastructures. *IEEE Transactions on Sustainable Computing*, 2019.
- [11] Ming Yan, Chien Aun Chan, André F Gyga, 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.
- [12] 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.
- [13] 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.
- [14] 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.
- [15] 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.
- [16] 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.
- [17] 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.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] Yunpeng Wang, Ping Lang, Daxin Tian, Jianshan Zhou, Xuting Duan, Yue Cao, and Dezhong Zhao. A game-based computation offloading method in vehicular multiaccess edge computing networks. *IEEE Internet of Things Journal*, 7(6):4987–4996, 2020.
- [22] 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.
- [23] 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.
- [24] 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.
- [25] 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.
- [26] 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.
- [27] 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.
- [28] 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.
- [29] Tayebah 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.
- [30] 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.
- [31] 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.
- [32] 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.
- [33] 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.
- [34] Zoltán Adám Mann. Secure software placement and

- configuration. *Future Generation Computer Systems*, 110:243–253, 2020.
- [35] 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.
 - [36] 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.
 - [37] 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.
 - [38] 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.
 - [39] 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.
 - [40] 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.
 - [41] 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.
 - [42] 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.
 - [43] 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.
 - [44] 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.
 - [45] 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.
 - [46] 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.
 - [47] 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.
 - [48] Yunbo Li, Anne-Cécile Orgerie, Ivan Roderio, 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.
 - [49] 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.
 - [50] 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.
 - [51] 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.
 - [52] 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.
 - [53] 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.
 - [54] Shahid H. Bokhari. On the mapping problem. *IEEE Trans. Computers*, 30(3):207–214, 1981.
 - [55] 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.
 - [56] 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.
 - [57] Hasan Ali Khattak, Hafsa Arshad, Saif ul Islam, Ghufuran 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.

- [58] 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.
- [59] Tuan Nguyen Gia and Mingzhe Jiang. Exploiting fog computing in health monitoring. *Fog and Edge Computing: Principles and Paradigms*, pages 291–318, 2019.
- [60] Davide Andrea Guastella and Evangelos Pournaras. Co-operative multi-agent traffic monitoring can reduce camera surveillance. *IEEE Access*, 2023.
- [61] 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.
- [62] 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.
- [63] 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.
- [64] 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.
- [65] 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.
- [66] 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.
- [67] 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.
- [68] 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.
- [69] 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.
- [70] Natarajan Gautam. *Analysis of Queues*. CRC Press, LLC, Boca Raton, Florida, United States, 2012.
- [71] 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.
- [72] 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.
- [73] Toni Janevski. *Traffic analysis and design of wireless IP networks*. Artech House, 2003.
- [74] Leila Ismail and Huned Materwala. Escove: Energy-sla-aware edge-cloud computation offloading in vehicular networks. *Sensors*, 21(15):5233, 2021.
- [75] 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.
- [76] 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.
- [77] 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.
- [78] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems*, 28(5):755–768, 2012.
- [79] Franz Christian Heinrich, Tom Cornebize, Augustin Degomme, Arnaud Legrand, Alexandra Carpen-Amarié, 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.
- [80] 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.
- [81] 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.
- [82] 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.

- [83] 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.
- [84] 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.
- [85] 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.
- [86] 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.
- [87] 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.
- [88] 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.
- [89] 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.
- [90] 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.
- [91] 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.
- [92] 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.

APPENDIX

APPENDIX A: SUPPLEMENTARY MATERIAL

Table V outlines the mathematical notations utilized throughout the paper.

TABLE V: 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_{im}^w	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')
$b_{jj'}^u$	Average uplink transmission rate of the link (j, j')
$b_{jj'}^d$	Average downlink transmission rate of the link (j, j')
F_j^p	Processing capacity of node j (MIPS)
F_j^m	Memory capacity of node j
F_j^s	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_i^p	CPU demand of service i (million instruction per request)
L_i^m	Memory demand of service i (bytes)
L_i^s	Storage demand of service i (bytes)
η_i	Desired QoS level for service i
h_i	Delay threshold for service i
V_{ij}^m	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_{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_{δ}^n	Power consumption of networking equipment
P_{δ}^s	Power consumption of server machines
P_j^i	Power consumption of node j in idle mode (watt)
P_j^a	Active power consumption of node j (watt)
P_j^c	Power consumption of cloud node c in idle mode (watt)
$P_j^{\bar{a}}$	Active power consumption of cloud node c (watt)
P_j^r	Ratio of renewable power supplied to fog node j (server or edge/core router)
P_c^r	Ratio of renewable power supplied to cloud node c (server or switch)
P_m^r	Ratio of renewable power supplied to access point m
θ_c	PUE of cloud center c
p_j^f	Power consumption of edge/core router j for data transfer (watt per byte)
p_m^f	Power consumption of access point m for data transfer (watt per byte)
P_{δ}^E	Non-renewable power consumption of fog-cloud infrastructure
R_c	Average carbon emission rate for electricity
C_j^p	Unit cost of processing at node j (\$ per million instructions)
C_j^s	Unit cost of storage at node j (\$ per byte per second)
C_j^m	Unit cost of RAM at node j (\$ per byte per second)
$C_{jj'}^d$	Communication cost of link (j, j') (\$ per unit bandwidth per second)
C_i^d	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_{δ}^p	Cost of processing for plan δ
O_{δ}^s	Cost of storage for plan δ
O_{δ}^m	Cost of RAM usage for plan δ
O_{δ}^c	Cost of communication for plan δ
O_{δ}^d	Cost of deployment for plan δ
O_{δ}^v	Cost of deadline violation for plan δ
O_{δ}^E	Cost of energy consumption for plan δ
O_{δ}^F	Cost of carbon footprint for plan δ

Table VI illustrates the changes in service provisioning cost components for the greedy and baseline approaches compared to EPOS, highlighting both improvements (indicated by +) and deteriorations (indicated by -).

Figure. 11 illustrates the coefficient of variance and service provisioning cost in optimized routes scenario. In this sce-

TABLE VI: Comparison of Costs and Percentage Differences

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

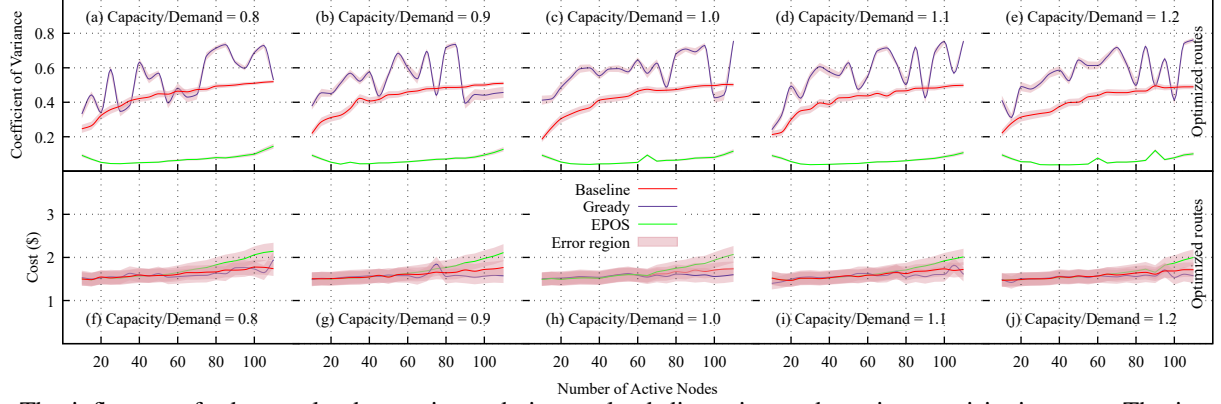


Fig. 11: The influence of edge-to-cloud capacity and size on load dispersion and service provisioning cost: The increase in capacity reduces costs and converges them across all approaches, the increase in the number of active nodes increases utilization variation in all methods.

nario, 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 ICT 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.