ADAPTO: Scaling and Offloading Cloud-Native Network Functions in Future Mobile Networks

Alessio Botta[1], Roberto Canonico[1], Annalisa Navarro[1], Giovanni Stanco[1], Giorgio Ventre[1], Antonio Buonocunto[2], Antonio Fresa[2], Enzo Gentile[2] Leonardo Scommegna[3], Enrico Vicario[3], Enzo Mingozzi[3,4], Antonio Virdis[4], and Marcello Cucurachi[5]

[1] Department of Electrical Engineering and Information Technology (DIETI), University of Naples Federico II, Naples, Italy
{a.botta, roberto.canonico,annalisa.navarro, giovanni.stanco,giorgio.ventre}@unina.it
[2] Ericsson Telecomunicazioni S.p.A., Pagani, Italy
{antonio.buonocunto,antonio.fresa,enzo.gentile}@ericsson.com
[3] Department of Information Engineering, University of Florence, Florence, Italy
{leonardo.scommegna,enrico.vicario}@unifi.it
[4] Department of Information Engineering, University of Pisa, Pisa, Italy
{enzo.mingozzi,antonio.virdis}@unipi.it
[5] Maticmind S.p.A., Naples, Italy
{marcello.cucurachi}@maticmind.it

Abstract. The advent of 5G networks has accelerated the adoption of cloud-native principles in telecommunications, enabling greater flexibility, scalability, and efficiency in network management. Central to this evolution is the integration of Network Function Virtualization (NFV), which transforms traditional hardware-dependent network functions into Virtual Network Functions (VNFs) composed of modular, interconnected microservices. While this approach offers significant advantages, it introduces challenges in resource allocation, VNF scaling, and service placement, particularly in Multi-Access Edge Computing (MEC) environments. This paper presents the ADAPTO framework, which provides an orchestration framework for optimizing computational and network resources across distributed edge and central clouds. We detail updated components within ADAPTO that enable dynamic resource monitoring, modeling of microservice dependencies using service call graphs, and decision-making for VNF scaling and placement. These innovations ensure optimized resource utilization, reduced latency, and enhanced reliability in cloud-native telecommunications networks.

Keywords: 5G and beyond· Network Function Offloading · Multi-access Edge Computing.

1 Introduction

Modern Multi-Access Edge Computing (MEC) environments in 5G networks pose significant challenges in computational and network resource allocation.

These challenges arise from the need to meet ultra-reliable low-latency communications (URLLC), high throughput, and scalability for diverse applications [1]. To address these demands, integrating Central and Edge Cloud infrastructures into a seamless Edge-Cloud Continuum has become essential. This integration provides the flexibility to offer high computational capacity and low-latency services for user equipment (UE), especially in the context of Network Function Virtualization (NFV).

NFV replaces traditional, hardware-based network functions with Virtual Network Functions (VNFs) deployed in virtualized environments. As telco networks adopt cloud-native principles, VNFs are increasingly designed using microservice architectures, enabling modularity, scalability, and agility. This granularity allows not only the independent placement and scaling of entire VNFs, but also the finer-grained control of individual microservices. However, managing these microservices introduces complexities due to their interdependencies - which often take the form of an intricate graph known as a *call graph* - resource constraints, and stringent latency requirements.

The ADAPTO framework, developed as part of the RESTART initiative under Italy's *National Recovery and Resilience Plan* (NRRP), addresses these challenges. Leveraging advancements in Software-Defined Networking (SDN) and NFV, ADAPTO optimizes the orchestration of computational and network resources in distributed cloud infrastructures. It dynamically adapts to fluctuating workloads and network conditions, ensuring efficient deployment and management of VNFs.

This paper contributes to the development of ADAPTO by presenting updated components that enhance the monitoring of computing resources and microservice interactions. These updates enable optimized scaling and placement of microservices, aligning with cloud-native principles to maximize resource utilization and minimize latency. The framework's key contributions include autonomous scaling of microservices, intelligent offloading between edge and central clouds, and adaptive backhaul selection to meet URLLC requirements. While previous work has explored the backhaul selection mechanism [2], further refinement is needed for the scaling and offloading components.

The remainder of this paper is organized as follows: Section 2 provides an overview of the system model. Section 3.1 details two telco cloud-native network functions and their microservice dependencies. Section 3.2 describes VNF scaling and placement techniques. Section 3.3 explains how service call graphs inform scaling and placement decisions. Section 4 provides a description of ADAPTO and its components, while Section 5 discusses conclusions and future directions.

2 System model

In this section, we briefly present the technological context in which the ADAPTO project provides its research contributions. Modern telco networks leverage technologies like Edge Computing, Multi-access Edge Computing (MEC) and cloud-native architectures to host computing capabilities across multiple sites, includ-

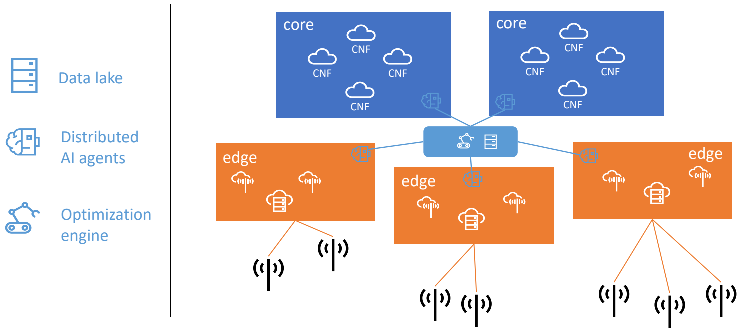

Fig. 1. ADAPTO System Model in the Edge-Cloud Continuum.

ing core data centers and edge locations near the radio access network (RAN). As shown in Figure 1, the computing resources deployed closest to the users are at the Edge layer, while the furthest computing resources are deployed at the Core (or Cloud) layer. Edge and Core layers are characterized by different amounts of available resources: typically fewer resources are deployed at the Edge and more at the Core. VNFs can be hosted either at the Edge or at the central cloud. The orchestration of the VNFs across the Cloud-Edge Continuum is possible thanks to the use of distributed agents whose main task is the monitoring of pertinent resources and the consequent scaling or placement of VNFs. These distributed agents are able to communicate and share data: these data are collected in a centralized data lake and represent the basis for the informed decision-making of the optimization engine.

3 Network Function orchestration in 5G and beyond

This section describes scaling and placement decisions for microservices in cloud-native network functions (NFs). First, we illustrate the composition of two NFs, showing how they are made up of microservices that can be shared, interact, or operate independently. Next, we discuss scaling and placement decisions, considering microservice dependencies, resource constraints and latency requirements. Finally, we introduce the service call graph, used to model these dependencies and optimize scaling and placement.

3.1 Network Function composition

Fig. 2 illustrates the microservice architecture for two core network functions: **Policy and Charging Control (PCC)** and **Policy Control Gateway (PCG)** in the modern cloud-native Ericsson telecom network [3]. These functions work together to enforce policies, manage user sessions, and handle charging aspects.

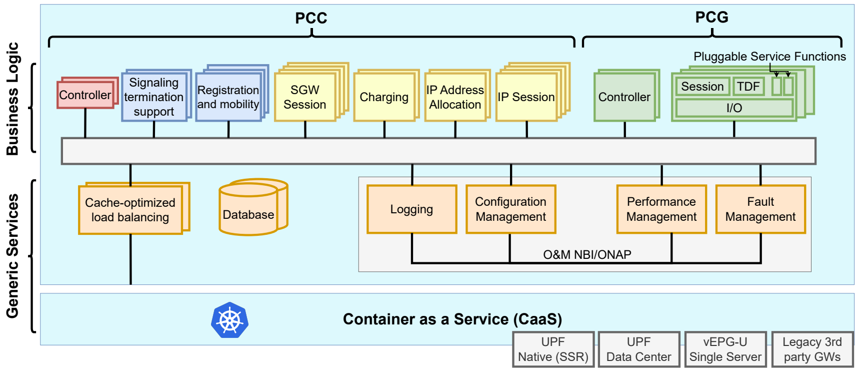

Fig. 2. 5G/6G microservice architecture for Network Function deployment in commercial solutions [7].

This architecture leverages microservices to decompose the functionality into individual, manageable, and scalable services.

The **PCC Microservices** are handling core functions like access, session, mobility and gateway control functions to support the new 5G use cases [4]. With functions like reduced signaling, adaptive paging algorithms, as well as the capability to provide service continuation to subscribers during network disturbance and network assurance with software probes, it includes a complete feature-set for MBB, Massive IoT, VoLTE and 5G NSA/SA. The **PCG Microservices** are focusing instead on user plane traffic processing and gateway function with advanced features like traffic and video optimization, NAT, firewall, software probes as well as the capability to include leading 3rd party applications [5].

In addition to specific microservices for PCC or PCG, **Generic Services microservices** are common services used by multiple NFs. Examples include the *Cache-Optimized Load Balancing* service, the *Database* for storing critical data such as user sessions and network states, and the *Fault Management* service for detecting and handling faults. These services provide cross-functional capabilities such as performance management, logging, and network configuration.

Each NF microservice can be **replicated** to meet network demands. This approach ensures high availability and fault tolerance, as additional instances of any service can be dynamically spun up as needed. Some components in the architecture are logically "linked", meaning they are tightly coupled or share similar functionality. These components are color-coded to indicate their close relationship. For instance, in the PCC section, services like *SGW Session*, *Charging*, and *IP Session* are depicted in yellow, reflecting their common role in session management, billing, and IP address allocation. Similarly, the *PCG* services, such as *Session*, *TDF*, and *I/O*, are grouped together, signifying their complementary roles in session management, traffic detection, and external network interactions.

[7] `https://events19.linuxfoundation.org/wp-content/uploads/2018/07/Ericsson-Cloud-native-tutorial.pdf`

3.2 Service scaling and placement

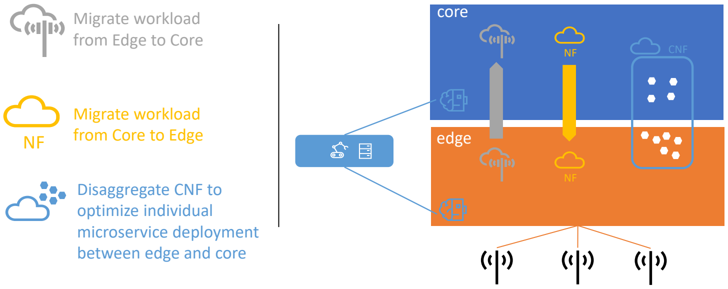
Migrate workload from Edge to Core
NF
Migrate workload from Core to Edge
Disaggregate CNF to optimize individual microservice deployment between edge and core
core
NF
CNF
edge
NF

Fig. 3. Network Function placement in the Edge-Cloud Continuum

Scaling and placement of Network Functions (NFs) are critical aspects that impact the network's performance, resource utilization, and energy efficiency [6]. These decisions must be dynamic and adapt to changing network conditions, ensuring efficient deployment of services and optimal use of resources.

The core layer handles high-computation tasks, while the edge layer supports localized services near UEs to minimize latency and backhaul traffic. Migration to the core (gray in Fig. 3) occurs when edge resources are constrained or centralized computational power is needed. Conversely, migration to the edge (yellow in Fig. 3) offloads tasks to reduce latency and enable localized processing, essential for latency-sensitive use cases like URLLC. Fig. 3 also depicts a finer-grained disaggregation strategy in which cloud-native functions are split into individual microservices. This disaggregation enables the selective deployment of microservices of a single NF at either the core or the edge. Such finer-grained control allows for better allocation of resources. Vertical scaling is performed at the granularity of the single microservice, allowing for precise resource adjustments. This approach to vertical scaling better aligns with energy efficiency objectives by minimizing unnecessary resource usage while maintaining service quality.

While dynamic scaling and placement are necessary to optimize resource utilization, it is equally important to consider the relationships between microservices when making scaling and placement decisions [7]. Microservices that are tightly coupled, such as those involved in session management or billing, should ideally be placed on the same site to minimize service completion time lowering network bandwidth consumption and latency.

3.3 Service call graph

The service call graph is a structured representation that captures the interactions and interdependencies between microservices within a service. Each node in the graph represents a microservice, and each edge denotes a communication

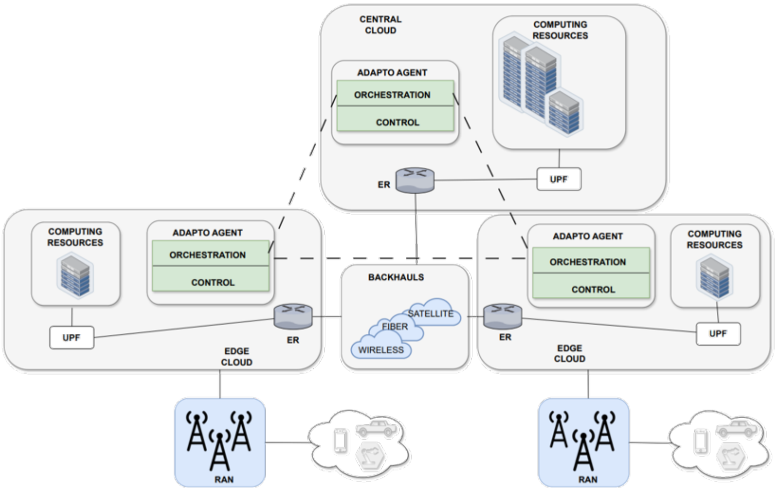

Fig. 4. ADAPTO distributed architecture with decentralized agents.

or data exchange between them. This graph serves as a foundational tool for understanding the operational relationships and data flows between microservices, enabling informed decisions about their deployment across sites. For example, if a call graph reveals a high degree of dependency or frequent and high-bandwidth data exchanges between two microservices, co-locating them within the same site can improve performance by minimizing communication delays and overhead.

A service call graph can be derived by using advanced monitoring and tracing techniques applied to the service infrastructure [8]. Network Traffic Analysis Monitoring tools can analyze network traffic between microservices to identify communication patterns and extract interdependencies. This method can provide real-time updates to the call graph, ensuring it reflects current service behavior. Besides giving indications on microservice placement, the call graph also guides vertical or horizontal scaling decisions. For example, a highly interconnected microservice causing a bottleneck can be prioritized for resource scaling.

4 ADAPTO distributed framework

In this section, we describe the ADAPTO distributed framework, focusing on its general architecture and showing how an ADAPTO agent performs orchestration and control actions in each site.

4.1 Architecture

Fig. 4 illustrates the reference architecture. The ADAPTO agents are deployed in both edge and central clouds. These agents manage the orchestration of NFs

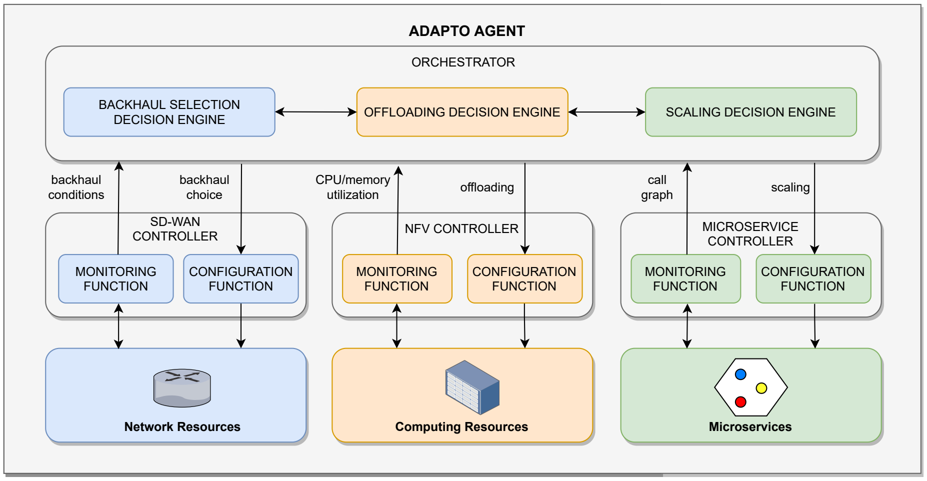

Fig. 5. Resource orchestration and control within a single ADAPTO agent.

and their microservices while controlling the underlying network and computing resources. The orchestration component of the ADAPTO agents focuses on dynamic resource allocation to meet Service Level Objectives (SLOs), while the control component enforces policies by monitoring device state, configuring them, and actuating the decision taken by the orchestration plane.

The architecture integrates diverse backhaul networks that connect central and edge clouds. Backhaul networks mix satellite, fiber, and wireless technologies, ensuring robust and flexible interconnectivity with high reliability and efficient data transfer. A Software-Defined Wide Area Network (SD-WAN) approach is employed to dynamically select the most suitable backhaul path, optimizing network quality of service (QoS) policies and responding to dynamic network conditions [9]. The edge routers (ERs) play a crucial role in interconnecting the various sites, while the User Plane Function (UPF) is used to handle packet forwarding [6]. Depending on the orchestration plane decisions, data can be directed to local computing resources at the edge or to the central cloud.

Within the edge deployment, the UPF manages traffic direction, either directing it to the edge or the central cloud through the ER. Different ERs are employed - situated at the edge or in the cloud - as in the SD-WAN paradigm [10]. These ERs are responsible for selecting the optimal backhaul among the available options (e.g. Satellite, Fiber, etc.) to connect the edge to the central cloud, or the edges with each other, aiming for a satisfactory latency value.

4.2 ADAPTO agent

Fig. 5 shows the detailed architecture of the devised ADAPTO agent. In the following, the main components are described:

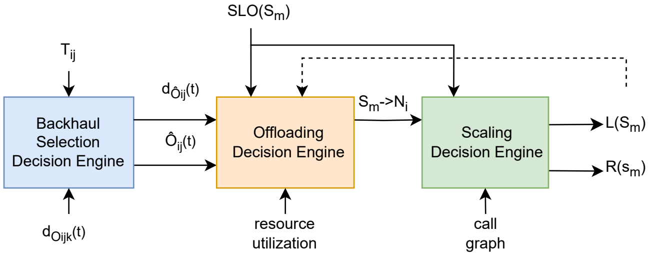

Fig. 6. Interactions in the ADAPTO **Orchestration Plane**. $d_{Oijk}(t)$: delay of k-th overlay O between site i and site j; T_{ij}: maximum delay threshold between site i and site j; $\hat{O}_{ij}$: chosen overlay between site i and site j; $d_{\hat{O}_{ij}}$: delay on the chosen overlay; SLO: Service Level Objective on task completion time; S_m: m-th service; s_n: n-th microservice; N_i: i-th site (can be edge cloud or central cloud); $L(S_m)$: completion time for service S_m; $R(s_n)$: allocated resources for microservice s_n.

- **Microservice Controller**: monitors the traffic between the microservices to retrieve the call graph, and implements the scaling decision it receives from the scaling decision engine.
- **Scaling Decision Engine**: decides about the scaling of the individual microservices composing the VNFs based on the call graphs monitored by the underlying monitoring function.
- **NFV Controller**: is tasked with the continuous monitoring of the resources at the edge, specifically tracking the percentage of usage of memory, CPU, and storage.
- **Offloading Decision Engine**: determines whether the NFs or the individual microservices that comprise the NFs should be offloaded to the central cloud if the edge resources become insufficient.
- **SD-WAN Controller**: monitors latency on available backhauls by employing probing mechanisms and configures ERs to select the backhaul that aligns with QoS requirements.
- **Backhaul Selection Decision Engine**: determines the backhaul to use to reach the central cloud when an offloading decision is triggered based on the backhaul monitoring.

4.3 Orchestration

The ADAPTO agent orchestration plane - whose interactions are depicted in Fig. 6 - integrates several key modules.

Backhaul selection decision engine plays a critical role in maintaining network performance by continuously monitoring network conditions and dynamically selecting the most suitable overlay between nodes. This module operates in near real-time and is based on Reinforcement Learning [11]. It considers the maximum allowable delay threshold (T_{ij}) and the latency of different overlays $d_{O_{ijk}}$ as inputs and outputs a geographical topology that specifies O_{ij} - the selected overlay between sites - and their associated delays $d_{O_{ij}}$.

Offloading decision engine Complementing the backhaul optimization, it focuses on allocating NFs or microservices composing them to specific sites to balance low-latency requirements with resource constraints. Initially, services are deployed on edge sites to satisfy latency-sensitive applications. However, in cases of resource overload at the edge, the module can offload services to the central cloud or, potentially, to other neighboring edge sites. The module takes as input the SLOs, the response time of the service (computed by the monitoring module of the microservice controller), the latency to reach the cloud (determined by the backhaul selection module), and the utilization metrics for edge CPU and memory. The functions of the offloading decision engine can also be extended to achieve energy-saving goals in the long term by exploiting traffic variability over time, e.g., peak traffic during working hours vs low traffic at night time. Such variability can be exploited to switch off part of the edge-based computing infrastructure during off-peak hours, thus saving energy. This can be done for example following a service-consolidation approach, wherein a selected set of computing resources is first emptied by migrating the running NFs to different servers, and then switched off to save energy. To back up this approach, the ADAPTO framework provides traffic prediction capabilities, which allows the offloading decision engine to predict near-future traffic variation and adjust the number of active servers, reducing energy consumption in the long term.

Scaling decision engine dynamically adjusts the resources allocated to each microservice within a node to meet SLO requirements. This module also takes advantage of the traffic monitoring between microservices to build a service call graph. Taking this as an input together with the service SLO, the output of this engine specifies the resource allocation for each microservice R_{s_m} ensuring efficient scaling decisions, and the task completion time L_{s_m} of the service S_m given the resources allocated to its microservices. The call graph is also sent to the offloading module, which takes it into account to decide which microservices are too tightly coupled to be deployed separately. The service placement module consists of a model-based decision engine that takes into account the compositional top-down approach to determine resource allocation as in [12].

5 Conclusion

In this paper, we introduced ADAPTO, a framework for managing and optimizing computational and network resources in distributed cloud-native telecom infrastructures. Designed to address the challenges posed by 5G and beyond networks, ADAPTO leverages advancements in Network Function Virtualization (NFV) and Software-Defined Networking (SDN) to enable the intelligent scaling, placement, and offloading of Virtual Network Functions (VNFs) across the Edge-Cloud Continuum. The detailed architecture of ADAPTO highlighted its capability to adjust to varying workload demands and network conditions, optimizing both computational and network resources. Key features such as dynamic backhaul selection, intelligent VNF offloading, and autonomous scaling underline its potential to meet the complex requirements of modern telecom networks. Future work will focus on an integrated evaluation of the components, particularly the scaling, offloading, and backhaul selection modules.

Acknowledgment

This work was partially supported by the European Union through the ADAPTO project, part of the RESTART program, NextGenerationEU PNRR, CUP E63C2 2002040007, CP PE0000001.

References

1. Fatima Salahdine, Tao Han, and Ning Zhang. 5G, 6G, and Beyond: Recent advances and future challenges. *Annals of Telecommunications*, 78(9):525–549, 2023.
2. Alessio Botta, Roberto Canonico, Annalisa Navarro, Giovanni Stanco, Giorgio Ventre, Antonio Buonocunto, Antonio Fresa, Vincenzo Gentile, Leonardo Scommegna, and Enrico Vicario. Edge to Cloud Network Function Offloading in the ADAPTO Framework. In *International Conference on Advanced Information Networking and Applications*, pages 69–78. Springer, 2024.
3. Ericsson. Ericsson enables efficient transformation to cloud native with new Compact Packet Core. https://www.ericsson.com/en/news/2024/12/ericsson-enables-efficient-transformation-to-cloud-native-with-new-compact-packet-core, 2024.
4. Ericsson. PCC. https://www.ericsson.com/en/portfolio/cloud-software-and-services/cloud-core/packet-core/cloud-packet-core/packet-core-controller.
5. Ericsson. PCG. https://www.ericsson.com/en/portfolio/cloud-software-and-services/cloud-core/packet-core/cloud-packet-core/packet-core-gateway.
6. Davit Harutyunyan, Rasoul Behravesh, and Nina Slamnik-Kriještorac. Cost-efficient placement and scaling of 5G core network and MEC-enabled application VNFs. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 241–249. IEEE, 2021.
7. Seyedeh Negar Afrasiabi, Amin Ebrahimzadeh, Azadeh Azhdari, Carla Mouradian, Wubin Li, Róbert Szabó, and Roch H Glitho. Joint VNF Decomposition and Migration for Cost-Efficient VNF Forwarding Graph Embedding. In *GLOBECOM 2023-2023 IEEE Global Communications Conference*, pages 2863–2869. IEEE, 2023.
8. Wilhelm Hasselbring and André van Hoorn. Kieker: A monitoring framework for software engineering research. *Software Impacts*, 5:100019, 2020.
9. Ayoub Mokhtari and Adlen Ksentini. SD-WAN for Cloud Edge Computing Continuum interconnection. *GLOBECOM*, 2024.
10. Junjie Wang and Lihong Zheng. SD-WAN: Edge Cloud Network Acceleration at Australia Hybrid Data Center. In *International Conference on Advanced Information Networking and Applications*, pages 659–670. Springer, 2022.
11. Alessio Botta, Roberto Canonico, Annalisa Navarro, Giovanni Stanco, and Giorgio Ventre. Adaptive overlay selection at the SD-WAN edges: A reinforcement learning approach with networked agents. *Computer Networks*, 243:110310, 2024.
12. Benedetta Picano, Riccardo Reali, Leonardo Scommegna, and Enrico Vicario. Elastic Autoscaling for Distributed Workflows in MEC Networks. In *International Conference on Advanced Information Networking and Applications*, pages 151–160. Springer, 2024.