Intelligent Service Provisioning in Fog Computing

Intelligent Service Provisioning in Fog Computing

Gaetano Francesco Pittalà
DEI - University of Bologna
Bologna, Italy
francesco.pittala@unibo.it

Walter Cerroni
DEI - University of Bologna
Bologna, Italy
walter.cerroni@unibo.it

***Abstract*—Fog computing is a distributed paradigm that extends cloud computing closer to the edge of the network, and even beyond that. By employing local resources, it enables quicker and more effective data processing and analysis. The optimization and automation of resource allocation, data processing, and job scheduling in the fog environment are made possible by the application of machine learning to Fog Computing Orchestration. It is also important, when working with the network computing models, to consider the XaaS paradigm, as it promotes the flexibility and scalability of fog services, bringing the concept of "service" into the foreground. Therefore, the need for a fog orchestrator enabling such characteristics arises, leveraging AI and the "service-centric" approach to enhance users' service fruition. The design and development of such an orchestrator will be the objective of the early-stage PhD project presented in this paper.**

***Index Terms*—Everything-as-a-Service, Fog Computing, Service Orchestration**

I. Introduction and Context

Modern telecommunication systems are meant to be distributed, flexible, scalable, and intelligent. They are required to dynamically adapt themselves to handle the amount of traffic coming from the ever-increasing number of connected devices. Furthermore, telecommunications equipment is expected to support agile and efficient provisioning of new services, which are highly diversified in terms of requirements and characteristics, to meet the heterogeneity of the service providers and users, guaranteeing their required Quality-of-Experience (QoE). This trend is characterized by increasingly critical needs in terms of networking and computing performance, depending on application context [1]. New-generation network services must fulfill very tight latency requirements to end-user or machine-to-machine applications, thus making latency one of the most important parameters in the performance evaluation process of the service providers. The advent of the Internet of Things (IoT) paradigm has made this trend even stronger, bringing a massive number of devices that have to exchange information on networks inside houses, workplaces, and cities.

In the last decade, Cloud Computing (CC) was consolidated in the telecommunications world as the leading paradigm for service provisioning. It allows providers to offer network services, computing resources, and storage space to users in a highly efficient way, by taking advantage of both hardware and software resources deployed in the network. However, due to cost-efficiency reasons, such resources usually reside in remote locations and are shared with a large number of users, strongly affecting communication latency. For this reason, new paradigms for service provisioning such as Fog Computing (FC) and Edge Computing (EC) have been gaining massive traction and are being thoroughly investigated. The rationale behind them is that a large portion of services can be offered by resources that are easily deployable in network segments closer to end-users or data sources. Therefore, from the point of view of latency, FC and EC can achieve significant improvements in comparison to CC thanks to such proximity. Although the definition might vary depending on the context, EC usually includes computing facilities such as small data centers located immediately after the access segment, while FC often includes heterogeneous fixed and mobile resources found between the cloud and the edge, or even beyond that. These paradigms are expected to revolutionize next-generation network systems, enabling service providers to cope with increasingly stringent constraints. Moreover, these paradigms (including CC, which is foreseen to remain a fundamental part of the system) will enable groundbreaking Quality-of-Service (QoS) levels in networks, while also shedding light on new interesting research areas, such as the Edge-to-Cloud-Continuum (ECC), representing the intention of network tenants and service providers in befitting from true end-to-end (E2E) management and control capabilities [2].

In order to provide the aforementioned capabilities, the world of networking is moving towards a service-centric approach, possibly applied to EC and FC environments. A service is an entity running at the network application layer and/or above, that provides data storage, manipulation, presentation, communication, or other capabilities to the users. A service network is a structure that brings together several entities to deliver a particular composite service.

II. Problem Statement

Although the services offered by EC/FC infrastructures are meant to be largely equivalent to those offered by their Cloud counterparts, they are inherently different, starting from their more distributed and dynamic nature [3]. Hence, it is quite natural to borrow the Everything-as-a-Service (XaaS) Cloud service model classification [4] and apply it to EC/FC scenarios as well, thus extending it towards a more flexible and dynamic environment. The XaaS model includes sub-paradigms that provide specific characteristics to the users, such as Infrastructure-as-a-Service (IaaS), Platform-as-

a-Service (PaaS), Software-as-a-Service (SaaS), and Function-as-a-Service (FaaS). However, an additional effort is required to redefine the usage of the paradigm in a dynamic environment where resource-constrained nodes are employed to provide one out of a set of supported services, based on their availability at the time when a user requests it.

In order to cope with the complexity of this process, pervasive adoption of Artificial Intelligence (AI) is expected to take place. AI is a broad term, encompassing an evolving field of computational algorithms that are designed to emulate human intelligence by learning from experience in the surrounding environment, as well as covering several branches such as natural language processing, robotics, computer vision, and Machine Learning (ML). These techniques are evermore relevant following the advent of new technologies, such as the aforementioned IoT, that produced an explosion of data volumes generated by an increasing number of applications. This is strongly impacting the evolution of distributed digital infrastructures for data analytics and the application of ML techniques to them. While data analytics used to be mainly performed on Cloud infrastructures, the rapid development of IoT infrastructures and the requirements for low-latency and secure processing have motivated the development of Edge analytics [5].

III. Related Work

Fog computing has the ability to process data more quickly and efficiently closer to the network's edge, hence lowering latency. Many methods have been put forth in the field of orchestration, including model-driven orchestration and service-oriented orchestration. These techniques are often constrained in terms of flexibility and scalability.

Given that the OpenFog [6] is one of the most well-known open-source platforms for fog computing, it is helpful to mention it briefly before digging into the literature on fog orchestration. OpenFog is the standard in the field of FC, it aims at enabling the development and deployment of intelligent IoT applications by providing a standard architecture and framework for distributed computing at the edge of the network. The OpenFog reference architecture, however, is primarily concerned with the Fog Node (FN) level, as the orchestration level details are only superficial, and new specifications in this area are anticipated. In the literature, most of the works on fog orchestration make their foundations on the OpenFog standard.

In [7] authors propose a novel architecture for fog computing orchestration called "Foggy". In order to fulfill the demands of fog applications, which in turn seek access to and use the resources and services made available by that infrastructure, Foggy manages the workload placement in the Fog environment. In particular, its framework is decomposed into three tiers, called "cloud tier", "edge cloudlets" and "edge gateways", ordered based on the proximity of the requested resources to the user (e.g. the edge gateways nodes are placed nearer to the user compared to edge cloudlets nodes, providing applications with lower latency times).

In [8] authors developed a prototype orchestrator architecture to prove key fog concepts. In their tests two Key Performance Indicators (KPIs) were considered: Time to Orchestrate measures the start time and how the system behaves when orchestrating different services in different setups (e.g. varying image size, image location, etc.); Opportunity Losses verifies if all the requirements at a point in time are satisfied and the orchestration is successfully instantiated; Time to scale and time to migrate both single microservices or the entire composite service.

In [9] authors devise an architecture to manage resources in the Fog using a hybrid approach. In the IoT and South-Bound Fog Levels, a distributed management of applications and services is proposed, applying choreography techniques to enable automated fast decision-making. A centralized approach to orchestrate applications and services taking advantage of global knowledge of the resources available in the network is suggested for the North-Bound Fog and Cloud Levels instead.

The aforementioned work defines and proposes novel models for service orchestration in fog computing, in particular underlying its distributed nature and the need for efficient monitoring and information about resources availability of both nodes and services. However, even if the orchestrator possesses and makes use of services' information (regarding needed computational resources, location, access rights, etc.), as well as nodes' information, it uses such information in order just to provide a decision on which node to deploy the service on. Moreover, these approaches do not take into account the nature of the services themselves, neglecting the specific requirements of different provisioning models (i.e. IaaS, PaaS, SaaS and FaaS).

In [10], instead, authors propose a service-centric approach, bringing the nature of the services on top of the orchestration functionalities. They propose the so-called "FORCH" orchestration system, which makes use of the information collected from the nodes (i.e. the available resources), and deploys the services requested by the users. In particular, the activation of the selected service depends on the nature of the service itself (i.e. the service provisioning model that has to be used) and the current status of the system. However, the weakness of FORCH resides in the decision-making process regarding service placement. Indeed, the allocation process is based on resource availability from different nodes, considering only the current CPU availability of the different ones. This underlines the need for intelligent service placement in FC orchestration architecture.

Nowadays, current efforts to automate and enhance the orchestration process using ML have produced encouraging results. These methods, however, have mostly been designed for centralized cloud settings and might not work well in fog computing scenarios. Nonetheless, ML has found broad application in multiple aspects of fog computing, from improving data processing and analysis to enhancing security and energy efficiency.

In particular, in [11], [12], [13], and [14] the authors focus on the topic of task offloading in vehicular fog networks,

proposing different solutions for optimizing energy efficiency, service latency, and performance in general.

In [15] the authors focus on the problem of load balancing in fog networks. In particular, they propose a novel dynamic resource allocation method based on Reinforcement Learning (RL) and genetic algorithm. This technique monitors the traffic in the network continuously, collects information about each server load, handles the incoming requests, and distributes them equally and dynamically among the available servers.

In [16] authors deal with the problem of powering FN with unpredictable sources of energy, such as wind/solar sources. They propose a RL technique to choose a server activation policy that ensures the minimum job loss probability.

Finally, in [17] and [18] authors deal with resource/application provisioning. In particular, in [18] authors propose a new RL-based agent that learns about the best resource allocation decisions, focused on reducing costs and energy consumption.

In the last couple of years, a lot of effort has been put into implementing different ML techniques in different working mechanisms of Fog Networks. However, a real service-centric approach has not been considered when dealing with the provisioning of services to users in an intelligent way. A lot of effort from the research studies has been driven toward particular use cases and a general solution to efficient service provisioning has not arisen.

IV. Orchestrator for Intelligent Service Provisioning in Fog Computing

The introduction of the OpenFog [6] standard, followed by the need for an energy-efficient, secure, flexible, and scalable service deployment, assess the importance of a general-purpose fog orchestrator enabled with ML to provide such skills.

The expected outcome of this work, indeed, is a new AI-based service-aware Fog Computing orchestration system, able to exploit ML algorithms for different purposes, such as decision-making for service allocation or energy-efficient service provisioning. This includes the design and development of the required software components as well as their testing and security validation. This would be also accomplished thanks to the guidelines provided by other reference architectures and models such as the aforementioned OpenFog standard. Furthermore, the development of this new system relies on the achievement of a vast number of skills along with the understanding of several concepts, algorithms, and models, such as the awareness of the newest network security flaws, the knowledge of the best-performing ML algorithms and the most innovative network models.

The system is envisioned to revolve around AI-based components for data analytics and informed decision-making, based on monitoring information and closed-loop feedback from the underlying infrastructure. To this aim, different ML patterns, algorithms, and techniques are expected to be applied, such as Deep-Reinforcement Learning (DRL), which makes use of neural networks to make decisions based on a previous training phase and on a reward system. Of course, the approach chosen to handle such heavy ML algorithms would be selected thanks to a meaningful study of other architectures, understanding and underlying their points of strength, and their centralized/distributed nature.

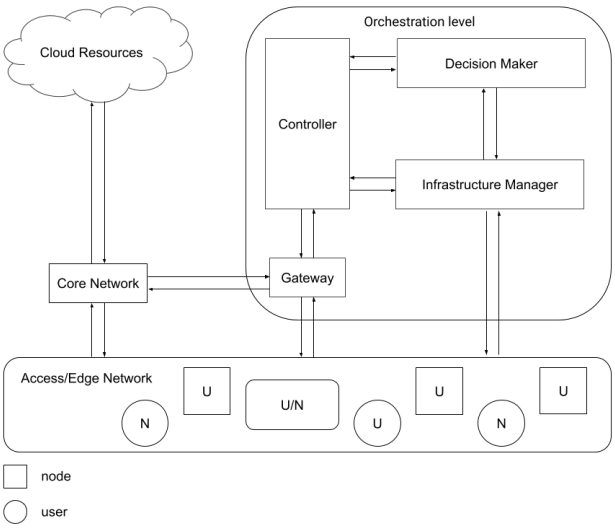

Fig. 1. System architecture.

Additionally, the orchestrator is intended to employ a "service-centric" approach, in which the service is considered the fundamental entity. The system is designed to map nodes to services, elevating the abstraction of services to the top hierarchical level in the software structure. This facilitates the implementation of the Everything-as-a-Service model on the proposed orchestration system.

The architecture of the proposed orchestration system is shown in Fig. 1 and is composed of multiple different macro-blocks needed to provide the desired functionalities. Although the orchestration level of the architecture is represented as a unique entity, each individual internal component can be unbundled and deployed in different locations of the network. Before introducing the different components at the orchestration level, it is worth describing the reference context the system will be working in. The orchestrator-level components will have the capabilities to interact with the edge/access network and the Cloud. The nodes and users can connect to either networks, communicating with the orchestrator through the former.

In Fig. 1 the presence of the Cloud is also highlighted, as it is an important component from the point of view of the system. The latter can pull contents (such as container images or configuration data) from the Cloud and use them to provide services to users, with the benefits of the Fog/Edge approach.

The following is a description of the functional components of the orchestration level, with details on the purpose and scope of each of them.

- **Controller**: it is the entity that oversees and coordinates the work of all other functional elements. It exchanges information with the Cloud and pulls data from it if necessary. It is also in charge of handling failures that

may occur in the system, taking control and trying to find a solution for them before interrupting the request being served.

- **Decision Maker (DM)**: it is the element where the ML algorithms are executed. As already underlined, AI is key in the orchestrator, so ML algorithms run by this entity enable the system to take the best fitting choices regarding where to deploy services and how to manage the users/nodes. It is interesting to notice that the connection between the Controller and the DM is needed, as it could be necessary to download newer and/or more effective algorithms for specific scenarios. On the other hand, the link between the DM and the Infrastructure Manager is necessary too, as information about how and where to allocate the services cannot be taken without the knowledge of the current configuration and status of hardware and software resources.
- **Infrastructure Manager (IM)**: it is the entity in charge of actuating the service deployment decisions taken by the other elements, as well as providing monitoring over the underlying resources. The IM can interact directly with hardware/software resources provided they have the necessary interfaces, and/or with another infrastructure management platform. For compatibility purposes, the IM element may have multiple instances, each specialized in the interaction with a different underlying infrastructure. The monitoring functionalities may require an additional agent component to be active at the node level.
- **Gateway (GW)**: it is the contact point between nodes and orchestrator. Through the GW the nodes can interact with the orchestrator requesting services. Conversely, additional information on nodes can be gathered through the GW by the system.

At FN level, ML techniques are expected to be implemented as well. As underlined, the orchestrator collects information from the nodes in order to provide suitable data to the DM and make a decision regarding which node to deploy the service on. Such data regards not only currently available resources on the nodes, but also information on resources consumption for previously deployed services. This enables the orchestrator, not only to reserve resources on the nodes taking into consideration the nature of the service itself and its requirements, but also to make decisions based on the expected energy consumption from the different nodes. From the point of view of energy efficiency, the nodes are expected to run agents able to learn how and when to accept service requests, optimizing energy usage. To enable intelligent management of resources it will be mandatory to understand also the relationship between energy consumption and providing a particular service to a user.

V. Conclusion

In conclusion, FC is a distributed paradigm that enhances cloud computing by enabling faster and more effective data processing and analysis. Orchestration in fog computing has been approached through different methods, including model-driven and service-oriented orchestration, and recent efforts have focused on a service-centric approach. However, the decision-making process regarding service placement still requires improvement, and current research is looking into the application of machine learning to fog computing orchestration. While there are challenges in implementing ML methods in fog computing, such as the need for distributed learning and limited resources, recent studies have shown promising results in optimizing energy efficiency, service latency, and performance in general. The development of a new system enabled with ML, in all the various aspects of service provisioning, will foster more efficient resource allocation and improved service delivery in fog computing environments, further enhancing the concept of XaaS.

References

[1] Y.-J. Ku *et al.*, "5g radio access network design with the fog paradigm: Confluence of communications and computing," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 46–52, 2017.

[2] K. Cao *et al.*, "A survey on edge and edge-cloud computing assisted cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 11, pp. 7806–7819, 2021.

[3] M. Iorga *et al.*, "Fog computing conceptual model," 2018-03-14 2018.

[4] P. Mell and T. Grance, "The nist definition of cloud computing, tech. rep." 2021.

[5] D. Rosendo *et al.*, "Distributed intelligence on the edge-to-cloud continuum: A systematic literature review," *Journal of Parallel and Distributed Computing*, 2022.

[6] OpenFog Consortium Architecture Working Group, "Openfog reference architecture for fog computing," *OPFRA001*, vol. 20817, p. 162, 2017.

[7] D. Santoro *et al.*, "Foggy: A platform for workload orchestration in a fog computing environment," in *2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2017, pp. 231–234.

[8] M. S. De Brito *et al.*, "A service orchestration architecture for fog-enabled infrastructures," in *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2017, pp. 127–132.

[9] K. Velasquez *et al.*, "Service orchestration in fog environments," in *2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*, 2017, pp. 329–336.

[10] G. Davoli *et al.*, "A fog computing orchestrator architecture with service model awareness," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2131–2147, 2021.

[11] Y. Wang *et al.*, "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 976–986, 2018.

[12] S. Vemireddy and R. R. Rout, "Fuzzy reinforcement learning for energy efficient task offloading in vehicular fog computing," *Computer Networks*, vol. 199, p. 108463, 2021.

[13] J. Zhao *et al.*, "Contract-based computing resource management via deep reinforcement learning in vehicular fog computing," *IEEE Access*, vol. 8, pp. 3319–3329, 2020.

[14] J. Shi *et al.*, "Priority-aware task offloading in vehicular fog computing based on deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 16 067–16 081, 2020.

[15] F. M. Talaat *et al.*, "A load balancing and optimization strategy (lbos) using reinforcement learning in fog computing environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 4951–4966, 2020.

[16] S. Conti *et al.*, "Battery management in a green fog-computing node: A reinforcement-learning approach," *IEEE Access*, vol. 5, pp. 21 126–21 138, 2017.

[17] M. Goudarzi *et al.*, "A distributed deep reinforcement learning technique for application placement in edge and fog computing environments," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[18] J. Santos *et al.*, "Resource provisioning in fog computing through deep reinforcement learning," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 431–437.