Fog and Cloud Computing Optimization in Mobile IoT Environments

José Carlos Ribeiro Vieira
josecarlosvieira@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa
Advisors: Prof. António Manuel Raminhos Cordeiro Grilo
Prof. João Coelho Garcia

Abstract. We introduce a xxxxx

Keywords: Cloud computing, fog computing, mobility, optimization, multi-objective

Table of Contents

1 Introduction

World is growing at a fast pace and so is data. Agility and flexibility of big data applications are gradually taking the form of the Internet of Things (IoT) and there's no doubt that it is a great resource. It comprises *things* that have unique identities and are connected to the Internet (e.g., vehicles, home appliances, wearable devices). The number of mobile devices are predicted to reach 11.6 billion by 2021, exceeding the world's projected population at that time (7.8 billion), where the subset of IoT ones are expected to be 929 million [1].

Although this kind of devices has evolved radically in the last years, battery life, computation and storage capacity remain limited. This means that they are not suitable for running heavy applications, being necessary, in this case, to resort to third parties.

Cloud computing (CC) is a resource-rich environment that has been imperative in expanding the reach and capabilities of IoT devices. It enables that clients outsource the allocation and management of resources (hardware or software) that they rely upon to the cloud. In addition, to avoid over- or under-provisioning, cloud service providers (CSPs) also afford dynamic resources for a scalable workload, applying a pay-as-you-go cost model. So, cloud computing is the on-demand delivery of compute power, database storage, applications, and other IT resources through a cloud services platform via the internet with pay-as-you-go pricing. This way, besides overcoming the aforementioned limitations, it also brings other advantages such as availability, flexibility, scalability, reliability, to mention a few.

Despite the benefits of using cloud computing, two main problems, linked to IoT applications, remain unresolved and they can not be underestimated. The first, and the most obvious, is the fact that cloud servers reside in remote data centers and, consequently, the end-to-end communication have long delays (characteristic of multi-hops transmissions over the Internet). Some applications, with ultra-low latency requirements, can't support such delays. Augmented reality applications that use head-tracked systems, for example, require end-to-end latencies to be less than 16 ms [2]. Cloud-based virtual desktop applications require end-to-end latency below 60 ms if they are to match QoS of local execution [3]. Remotely rendered video conference, on the other hand, demand end-to-end latency below 150 ms [4]. On the other hand, the constantly growing number of IoT devices raises the other problem. The basic premise is that as the number of connected devices increases, the bandwidth required to support them becomes too large for centralized processing (i.e. CC). As a fallout of this sense-process-actuate model, where the processing is done in the cloud, there are two immediately apparent options: (1) increase the number of centralized cloud data centers, what will be too costly, and (2) get more efficient with the data sent to the cloud.

To overcome this limitations, the solution that has already been proposed is to bring the cloud closer to the end devices, where entities such as base-stations would host smaller sized clouds. This idea has brought the emergence of several computing paradigms such as cloudlets [5], fog computing [6], edge computing [7], and follow me cloud [8], to name a few. However, they all share the same goal, achieve the option number (2), doing more processing in the things-to-cloud continuum instead of in the cloud.

Fog computing is a new computing architecture that aims to enable computing, storage, networking, and data management not only in the cloud, but also along the cloud-to-thing path as data traverses to the cloud. Fog nodes, also known as fog servers or cloudlets (smaller sized clouds with lower computational capacity), are virtual-machine (VM) based, which means that they promote flexibility and mobility. They can be placed close to IoT source

nodes, due to low hardware footprint and low power consumption. This allows latency to be much smaller, through geographical distribution, compared to traditional cloud computing and still cut off a significant amount of core network traffic. Nevertheless, cloud is still more suitable than fog for massive data processing, when the latency constraints are not so tight. So even though, fog computing has been proposed to grant support for IoT applications, it does not replace the needs of cloud-based services. In fact, fog and cloud complement each other, and one cannot replace the need of the other. Together, they offer services even further optimized to IoT applications. Moreover, Internet connectivity is not essential for the fog-based services to work, what means that services can work independently and send necessary updates to the cloud whenever the connection is available [9].

1.1 Motivation

Despite the benefits that fog promises to offer such as low latency, heterogeneity, scalability and mobility, the current model suffer from some limitations that still require efforts to overcome them.

There is lack of support for mobile fog computing. Most of the existing literature assumes that the fog nodes are fixed, or only considers the mobility of IoT devices [9]. Less attention has been paid to mobile fog computing and how it can improve the QoS, cost, and energy consumption. For instance, it does not foresee that a bus could have computational power; as a cloudlet, it cloud provide offloading support to elements (i.e. IoT devices and other cloudlets) inside and outside it. The same could be applied to cars that are nowadays getting increasingly better in terms of computational power. Both would be extremely useful in order to enhance the resources and capabilities of fog computing, for example, in large urban areas where traffic congestion is common or when they are parked. Apart from QoS enhancement, this would reduce the implementation costs since it would no longer require such computational power in the roadside cloudlets. Also, the costs in terms of latency to the client would be minimized. On top of that, it would minimize the energy consumption of mobile devices once the access points where they are connected may be even closer.

Another limitation of fog computing is to take into account few parameters in the decision making of migration. Most of the existing schemes that are proposed for fog systems, such as offloading, load balancing, or service provisioning, only consider few objectives (e.g., QoS, cost) and assume other objectives do not affect the problem [9]. Fog servers are less powerful than clouds due to the high deployment cost. If many requests are made to the same fog node at the same time, it will not have enough computational and storage power to give a prompt response. So it raises the question: should a service currently running in one fog node be migrated to another one, and if yes, where? While conceptually simple, it is challenging to make these decisions in an optimal manner. Offloading tasks to the next server seems to be the solution, however, migrate the VM that was initially one-hop away from the IoT device to a multi-hop away server, will increase the network distance. Consequently, raises the end-to-end latency and the bandwidth usage by the intermediate links. Besides, this decision still has to take into account the cost for both the client (e.g., migration time, computational delay) and the provider (e.g., computing and migration energy). Ignoring some of these parameters can lead to wrong decisions, what will both violate latency constraints of user's application and damage or defeat the credibility of fog computing.

1.2 Objectives

Summing up, this work intend to tackle two of the current limitations which are, to the best of our knowledge, untreated problems in the literature. One is to provide mobility support in fog computing environments, not exclusively to the end devices but also to the fog nodes and the other is to achieve multi-objective fog system design. These objectives shall be implemented in a toolkit allowing the simulation of resource management techniques in IoT, and mobile fog computing environments. In order to achieve the aforementioned goals, this work must follow the sequence of tasks presented below.

- Start with analysis of current mobility approaches publicly available with respect to IoT nodes or cloudlets;
- Propose mobile fog computing, where fog nodes can move, provisioning a method to keep the service always-available for IoT nodes;
- Analyze the most suitable optimization algorithms for fog computing and select the most appropriate ones;
- Design of mobility-aware task offloading when fog nodes are mobile, taking into account the related costs to the client (i.e. migration, communication and processing delay) and discovery;
- Propose a variation to achieve multi-objective (i.e. QoS, QoE, cost, handover, load balancing, energy, bandwidth and VMs or virtual objects migration);
- Study the current open source simulators for fog computing environments;
- Implement the proposed algorithms in the simulator and compare them.

The remainder of the document is structured as follows. Section 2 xxx. Section 3 describes xxxx. Section 4 defines the xxx. Finally, Section 5 presents xxxx and Section 6 xxxx.

2 Related Work

The solution proposed in this document leverages knowledge obtained from studying several concepts and systems from the current state-of-the-art. In this section an overview of those concepts and systems will be given, stating for each of them their advantages and disadvantages.
This section is structured as follows. Section 2.1 presents different methods to push intelligence and computing power closer to the source of the data and why this work adopted fog computing for this purpose. Section 2.2 presents mobility-aware systems xxx. Section 2.4 xxx. 2.5 xxx. 2.6 xxx. Section 2.7 xxx.

2.1 Computing paradigms

In what concerns about standardizing fog computing, there is a lack of unanimity. As aforementioned, fog has been variously termed as cloudlets, edge computing, etc. Different research teams are proposing many independent definitions of fog (and fog-related computing paradigms). As there is a research gap in the definitions and standards for fog computing, this work follows the definitions that Ashkan Yousefpour et al. [9] present. Below, are described some of the paradigms that were raised in order to bring cloud closer to the end devices, as well as their pros and cons. As a conclusion we show why fog computing is the natural platform for IoT.

2.1.1 Mobile computing

Mobile/nomadic computing (MC) is characterized by the processing being performed by mobile devices (e.g., laptops, tablets, or mobile phones). It raises to overcome the inherent limitations of environments where connectivity is sparse or intermittent and where there is low computing power. As this model only uses mobile devices to provide services to clients, there is no need for extra hardware, once they already have communication modules such as Bluetooth, WiFi, ZigBee, etc. As already mentioned, mobile devices have evolved in recent years, however their resources are more restricted, compared to fog and cloud. This computing paradigm has the advantage of being characterized by a distributed architecture, once mobile machines do not need a centralized location to operate. The disadvantages of MC are mainly due to their hardware nature (i.e. low resources, balancing between autonomy and the dependency of other mobile devices; characteristic that prevails in all distributed architectures) and the need of mobile clients to efficiently adapt to changing environments [10]. MC alone may not be able to meet the requirements of some applications. On the one hand it is limited due to autonomy constraints and in the order hand low computational power and low storage capacity further restricts the applications where this paradigm is feasible. For instance it is unsuitable for applications that require low-latency or that generates large amounts of data that needs to be stored or processed. Nonetheless, MC can use both fog and cloud computing to enhance its capacities, not being restricted to a local network; expanding the scope of mobile computing and the number of applications where it can be used.

2.1.2 Mobile Cloud Computing

Cloud and fog computing, as mentioned in 2.1.1, are key elements for validate the importance of MC. This interaction between them results in a new paradigm, called mobile cloud computing (MCC). MCC, differs from MC in a sence that mobile applications can be partitioned at runtime so that computationally intensive components of the application can be handled through adaptive offloading [11], from mobile devices to the cloud. This characteristic increases the autonomy of mobile devices (i.e battery lifetime), as both the data storage and data processing occur outside of the mobile device. Also, it enables a much broader range of mobile subscribers, rather then the previous laptops, tablets, or mobile phones. Opposed to resource-constrained in MC, MCC has high availability of computing resources, scaling the type of applications where it is possible to use, for instance, augmented reality applications. Unlike MC, MCC relies on cloud-based services, where its access is done through the network core by WAN connection, which means that applications running on these platforms require connection to the Internet all the time. On the one hand, both MCC an MC suffer from the intrinsic characteristics of mobility, such as frequent variations of network conditions, intensified under rapid mobility patterns. On the other hand, in MCC, even if the mobile devices remain fixed, it suffers from the inherent disadvantage of using cloud-based services (i.e. communication latency), which makes it unsuitable for some delay-sensitive applications.

2.1.3 Mobile ad hoc Cloud Computing

In some scenarios there exists lack of infrastructure or a centralized cloud, so implement a network based on MCC may not always be suitable. To overcome dependence on an infrastructure, raises mobile ad hoc cloud computing (MACC). It consists on a set of mobile nodes that form a dynamic and temporary network through routing and transport protocols. These nodes are composed by mobile ad hoc devices which may continuously join or leave the network. In order to counteract the, aforementioned, characteristics inherent to this type of networks, and unlike MC, a set of ad hoc devices may form a local cloud that can be used in the network for purposes of storage and computation. As mobile ad hoc networks (MANET), it can play a big part in use cases such as disaster recovery, car-to-car communication, factory floor automation, unmanned vehicular systems, etc. Although it does not rely in external cloud-based services as MCC does, which mitigates the latency problem, it shares some of the limitations inherent to MC and ad hoc networks such as the power consumption constraints. Moreover, the formed local cloud may still be computationally weak and as both network and cloud are dynamic it is more challenging to achieve an optimal performance. For instance, as there is no infrastructure, mobile devices are also responsible for routing traffic among themselves.

2.1.4 Edge Computing

Edge computing (EC), enhances their capabilities (i.e. management, storage, and processing power) by connected devices at the edge of the network. Edge computing is located in the local IoT network, being ideally at one hop away from the Iot device and at most located few hop away. Open Edge Computing defines edge computing as computation paradigm that provides small data centers (edge nodes) in close proximity to the users, enabling a dramatic improvement in customer experience through low latency interaction with compute and storage resources just one hop away from the user [12]. As in EC the connected devices do not

have to wait for a centralized platform to provide a requested service, nor are so limited in terms of resources as in the traditional MC, their service availability is relatively high. Nonetheless EC has some drawbacks. As latency, in this context, is composed by three components: data sending time, processing time and result receiving time, even though the communication latency is negligible, processing time may not be. This computing paradigm only uses edge devices, and their computation and storage power may be poor (e.g., routers, switches). Consequently, this processing latency may still be too high for some applications with low latency requirements.

OpenFog Consortium states that fog computing is often erroneously called edge computing, but there are key differences [13]. Although they have similar concepts, edge computing tends to be limited to the edge devices (i.e. located in the IoT node network), excluding the cloud from its architecture. Whereas, fog computing is hierarchical and unlike EC, it is not limited a local network, but instead it provides services anywhere from cloud to things. It is worth noting that the term edge used by the telecommunications industry usually refers to 4G/5G base stations, radio access networks (RANs), and internet service provider (ISP) access/edge networks. Yet, the term edge that is recently used in the IoT landscape refers to the local network where sensors and IoT devices are located [9].

2.1.5 Multi-access Edge Computing

Analogously, MCC is an extension of MC through cloud computing, as multi-access edge computing (MEC) is an extension of MC through EC. ETSI defines MEC as computation paradigm that offers application developers and content providers cloud-computing capabilities and an IT service environment at the edge of the network. This environment is characterized by ultra-low latency and high bandwidth as well as real-time access to radio network, information that can be leveraged by applications [14]. In MEC, operators can open their RAN edge to authorized third parties, allowing them to deploy applications and services towards mobile subscribers through 4G/5G base stations. The first approach to the edge of a network meant the edge of a mobile network, hence the name mobile edge computing. As MEC research progressed, was noticed that the term leaves out several access points that may also construct the edge of a network. Thus, prompted the change from mobile edge computing to multi-access computing in order to reflect that the edge is not solely based on mobile networks [15]. Now it include a broader range of applications beyond mobile device-specific tasks, such as video analytics, connected vehicles, health monitoring, augmented reality, etc.

MEC supports low-latency applications once it benefits from real-time radio and network information. Similar to edge computing, MEC can operate with little to no Internet connectivity and use small-scale data centers with virtualization capacity to provide services. Unlike EC, MEC establishes connectivity through a WAN, WiFi, and cellular connections, whereas edge computing generally can establish any form of connectivity (e.g., LAN, WiFi, cellular). MCC focuses on the relationship between cloud service users (on mobile devices) and cloud service providers, whereas research in MEC focuses on (RAN-based) network infrastructure providers. MEC allows edge computing to be accessible to a wide range of mobile devices with reduced latency and more efficient mobile core networks [16].SDN allows for virtual networking devices to be easily managed through software APIs [17], and NFV allows for reduced deployment times for networking services through virtualized infrastructure. MEC is expected to benefit significantly from the up-and-coming 5G platform as it

allows for lower latency and higher bandwidth among mobile devices, and it supports a wide range of mobile devices with finer granularity.

2.1.6 Cloudlet Computing

Cloudlet computing is another direction in mobile computing that aims to bring cloud closer to end devices through the use of cloudlets. M. Satyanarayanan et al. states that a cloudlet is a trusted, resource-rich computer or cluster of computers that's well-connected to the Internet and available for use by nearby mobile devices [18]. Cloudlet is, as the name suggests, a smaller sized clouds with lower computational capacity. It can be seen as a "data center in a box", where mobile users can exploit their virtual machines (VM) to rapidly instantiate customized-service software in a thin client fashion. This way, it is possible to offload computation from mobile devices to VM-based cloudlets located on the network edge. Through those VMs, cloudlets are capable of providing resources to end devices in real time over a WLAN network. The relatively low hardware footprint, results in moderate computing resources, but lower latency, energy consumption and higher bandwidth compared to cloud computing. The characteristics of this computing paradigm make possible to handle applications with low-latency requirements, supporting real-time IoT applications. Y Jararweh et al. [19] propose an architecture mobile-cloudlet-cloud, where they present three reasons which indicate that even though cloudlets are computationally powerful, they still need a connection to the cloud and its services: (1) Heavy non real time jobs might process in the enterprise cloud whiles the real time ones processed by the cloudlet, (2) Accessing a file stored in the Enterprise Cloud, (3) Accessing some services that are not available inside the Cloudlet. Although cloudlet computing fits well with the mobile-cloudlet-cloud architecture, fog computing offers a more generic alternative that natively supports large amounts of traffic, and allows resources to be anywhere along the thing-to-cloud continuum.

2.1.7 Mist Computing

Mist computing emerges to push IoT analytics to the extreme edge. This computing paradigm is an even more dispersed version of fog. That means locating analytics tools not just in the core and edge, but at the "extreme edge" - actually in many of the devices (the IoT devices themselves; the sensor and actuators) [20]. It has been proposed with future self-aware and autonomic systems in mind [21]. Therefore, mist computing can be seen as the first computing location in the IoT-fog-cloud continuum. Mist computing extends compute, storage, and networking across the fog through the things. This decreases latency and increases subsystems' autonomy. The challenge with implementing mist computing systems lies in the complexity and interactions of the resulting network, which must be managed by the devices themselves as central management of such systems is not feasible.

2.1.8 Fog computing

Fog computing bridges the gap between the cloud and IoT devices by enabling computing, storage, networking, and data management on the network nodes within the close vicinity of IoT devices. OpenFog Consortium [22] defines fog computing as "a horizontal system-level architecture that distributes computing, storage, control and networking functions closer to the users along a cloud-to-thing continuum." The decentralized nature of fog computing allows devices to either serve as fog computing nodes themselves (e.g. a

car acts as a fog node for on-board sensors) or use fog resources as the clients of the fog (end device). Compared to the cloud, fog computing offers moderate computing capacity and low power consumption. Clouds are composed by large datacenters whereas fog uses small servers, routers, switches, gateways, set-top boxes or access-points (AP). Fog has lower hardware footprint, so it can be closer to the end devices. Fog computing can be accessed through connected devices from the edge of the network to the network core, whereas cloud computing must be accessed through the network core. Moreover, continuous Internet connectivity is not essential for the fog-based services to work. That is, the services can work independently with low or no Internet connectivity and send necessary updates to the cloud whenever the connection is available. Cloud computing, on the other hand, requires devices to be connected when the cloud service is in progress. There are clear differences and trade-offs between cloud and fog computing, and one might ask which one to choose. However, fog and cloud complement each other; one cannot replace the need of the other. By coupling cloud and fog computing, the services that connected devices use can be optimized even further.

2.1.9 Other Similar Computing Paradigms

- **Micro Data Center** (MDC) - Cloudlet são por vezes referidas como MDC. Um MDC pode ser um edge node ou uma cloudlet que é implementada entre os dispositivos IoT e a cloud.
- **Cloud of Things** (CoT) - Parecido a mist computing, no entanto em mist a computação é feita nos dispositivos IoT e não necessariamente numa cloud de pooled resources. Em CoT, a computação é feita sobre a cloud que é formada por pooled resources de IoT devices.
- **Edge Cloud** - Quando falamos de cloud computing, falamos de "core" ou "distant" clouds que estão distantes do end users que são responsáveis pela computação "pesada". Ao contrário, "edge" clouds são mais pequenas e estão mais próximas. Edge cloud é parecido a edge computing. Edge cloud extende capacidades da cloud na edge aproveitando os compute nodes (user ou operator-contributed) na edge da rede. À semelhança do fog, em edge clouds a habilidade de correr uma aplicação numa forma coordenada tanto na edge com na cloud é prevista. Edge clouds são nós na edge como micro data centers, cloudlets, e MEC.

2.1.10 Concluding Remarks

As the above mentioned, there are some other similar computing paradigms such as follow me cloud (FMC), follow me edge (FME), Follow Me edge-Cloud (FMeC), etc. However, this state-of-the-art had as objective to investigate the most treated ones in the literature. The purpose was to understand their characteristics and identify where to tackle the current limitations which oppose the deployment of delay-sensitive IoT systems in mobile environments.

The previous discussion about fog computing and related paradigms demonstrate the importance of understanding the characteristics of these platforms in the changing IT landscape. As demonstrated by the strength and weaknesses attributed to these computing paradigms, some paradigms may be better suited for a particular use case than others. Even so, fog computing is suited for a large number of use cases in the current landscape of IoT and connected devices. The versatility of fog computing makes it suitable for many

cases of data-driven computing and low-latency applications, even though it may not be suitable for a few extreme applications, such as disaster zones or sparse network topologies where ad hoc computing (e.g., MACC) or extreme edge clouds (e.g., mist, CoT) may be a better fit. Nonetheless, fog computing is considered a more general form of computing when compared to other similar paradigms (e.g., EC, MEC, cloudlet), because of its comprehensive definition scope, generality, and extensive presence along the thing-to-cloud continuum. Tables 2 and 3 summarize these characteristics. Fog computing offers a bright future for an open-standards environment of connected devices, as it is evident by IEEE Standard's adoption of OpenFog Reference Architecture [69]. There does not yet exist a globally unanimous distinction between fog computing and related computing paradigms, such as edge computing, mist computing, and cloudlets across researchers and industries, as shown in the previous sections of this paper. We attempt in this survey paper to clarify the distinctions between fog computing and the related computing paradigms. A comparison of the underlying infrastructure of fog computing and its related computing paradigms from the networking perspective is shown in Fig. 7. In the rest of this paper, we will mainly survey and discuss the recent literature on fog computing, but mention the studies on other related computing paradigms that could be easily extended or directly applied in fog.

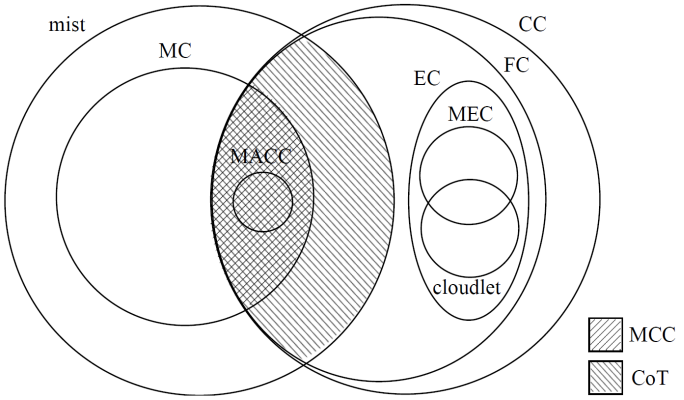

Fig. 1: A classification of scope of fog computing and its related computing paradigms.

DESCRICAO DA ARQUITETURA DE FOG COMPUTING which is the most typical one; a hierarchical, bi-directional computing infrastructure which consists in: a cloud, several cloudlets located at the access points (i.e. at the edge of the network) and several end devices. In this architecture, end devices communicate with cloudlets and cloudlets communicate with clouds. These cloudlets, which are capable of provide computing and storage services, can communicate with each other to perform data and process management in order to support application requirements, and to exchange fog control/management data (such as user device and application state).

In fog computing, processing and storage capacity is one hop away from the data production/consumption, which can benefit different types of applications

– Applications with low latency requirements, such as pedestrian and traffic security, surveillance, applications for vision, hearing, or mobilityimpaired users, online gaming,

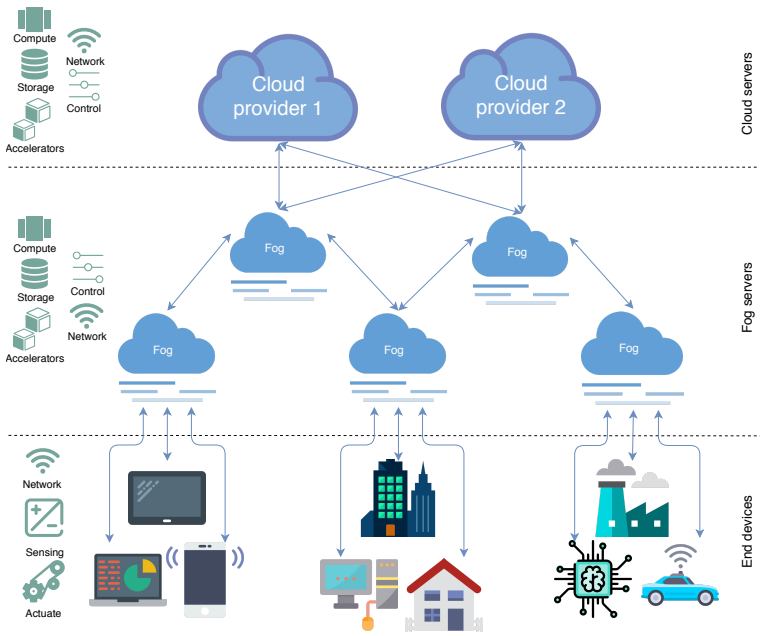

Fig. 2: Typical architecture of fog computing.

augmented reality, and tactile computing can benefit from lower latencies because of a single hop connection to a cloudlet.

– Raw data collected by many devices often does not need to be transferred to the cloud for longterm storage: data can be processed, filtered, or aggregated to extract knowledge and produce reduced data sets, which in turn are to be stored; or it can be processed and utilized right-away to other edge devices in the so-called sensor/actuator loop. In both cases, the fog computing paradigm can reduce network traffic from the edge to data centers.

Cloudlets can provide reduced latencies and help in avoiding/reducing traffic congestion in the network core. However, this comes at a price: more complex and sophisticated resource management and scheduling mechanisms are needed. This raises new challenges to be overcome, e.g., dynamically deciding what, when, and where (device/fog/cloud) to carry out processing of requests to meet their quality of service requirements. Furthermore, with smart and wearable devices, such mechanisms must incorporate mobility of data sources and sinks in the fog. Traditional resource management and scheduling models for distributed systems do not consider mobility and timeliness of data production and consumption in the resource management and allocation process. Fog computing scheduling must bring users location to the resource allocation policies to uphold the benefits of fog computing proximity to the user.

2.2 Mobile Fog Computing

Mobile Fog Computing is the ability to provide mobility support for both IoT and Fog nodes. This means that users holding end devices, can outsource the allocation and management of resources to the fog infrastructure, where both iot and fog nodes are able to move, ensuring that QoS requirements of the IoT applications are met.

Similar to cloud computing, fog uses the concept of virtualization. Virtualization is a vital technology at different levels. For instance, it is important to provide: (1) isolation between untrusted user-level computations, (2) mechanisms for authentication, access control, and metering, (3) dynamic resource allocation for user-level computations, (4) the ability to support a very wide range of user-level computations, with minimal restrictions on their process structure, programming languages or operating systems, (5) mobility, migration and tasks offloading mechanisms, (6) power efficiency, (7) fault tolerance.

IoT applications may span many different operating systems and application environments (e.g., Android, iOS, Linux, Windows), as well as diverse approaches to partitioning and offloading computation. There is churn in this space from new OS versions, patches to existing OS versions, new libraries, new language runtime systems, and so on. In order to cloudlets can support all these variants, it is introduced a level of abstraction that cleanly encapsulate this messy complexity in a VM. Also, it enables VMs to coexist in a physical server (host) to share resources. Meanwhile, in fog computing the use of VMs is also crucial to provide mobility. As the end devices become more distant from the current access point, either by user or cloudlet movement, its application needs to be migrated to the most favorable access point, towards meeting its QoS (e.g., latency requirements) and improve quality of experience (QoE). This migration can be achieved through migration capabilities of VMs, which can be used to move applications and data through cloudlets according to device mobility.

In this context, Luiz F. Bittencourt et al. [23] took into account the same architecture shown in Fig. 2. With the use of two applications (electroencephalography (EEG) tractor beam game to test near-real-time applications and video surveillance / object tracking application for delay-tolerant applications) to test tree different scheduling strategies (Concurrent, the First Come-First Served (FCFS), and the Delay-priority strategies), and check how scheduling decision and the change of cloudlet by the players impact the network traffic and the delays.

MM Lopes et al. [24] discuss resource allocation in fog computing in the face of users' mobility, where mobility is achieved through migration of virtual machines between cloudlets. They present a new migration technique composed of two modules: migration policy which defines when the user VM should be migrated, considering aspects such as the user's speed, direction and geographical position and migration strategy, the destination cloudlet, and how the migration is performed. This work had the objective of study the impact of different migration strategies in the latency with users' mobility.

[325] 2017 **Router-based brokering for surrogate discovery in edge computing.** This paper examines the problem of discovering surrogates, which are micro-clouds, fog nodes, or cloudlets, used by client devices to offload computation tasks in a fog computing environment. In order to enable the discovery and selection of available surrogates, the authors

propose a brokering mechanism in which available surrogates advertise themselves to the broker. The broker receives client requests and considers a number of attributes such as network information, hardware capabilities, and distance to find the best available surrogate for the client. The proposed mechanism is implemented on off-the-shelf home routers. They look at the problem of surrogate discovery in the context of an urban area, where they are faced with a high mobility of devices and users. Multiple brokers are interconnected using Distributed Hash Tables (DHTs) in order to exchange information. In addition, their approach introduces only a small overhead on the devices (routers) and therefore does not impede their normal function.

2018 **Dynamic Mobile Cloudlet Clustering for Fog Computing.** Fog Computing is one of the solutions for offloading the task of a mobile. However the capability of fog server is still limited due to the high deployment cost. In this paper, is proposed a dynamic mobile cloudlet cluster policy (DMCCP) to use cloudlets as a supplement for the fog server for offloading. The main idea is that by monitoring each mobile device resource amount, the DMCCP system clusters the optimal cloudlet to meet the requests of different tasks from the local mobile device.

Although several studies were already done in order to provide mobile support for IoT devices, the purpose of this study is to support mobile fog computing, once fog nodes can be anything in the path that connects things to the cloud. This distributed middle tier, in the 3-tier architecture (things-fog-cloud), can use as fog nodes any physical device that has facilities or infrastructures that can provide resources and visualization capabilities. This, may include movable fog nodes, such as cars, buses, unmanned aerial vehicles (UAVs), etc. The importance of mobile fog nodes cannot be overlooked, once they may represent a way to offload fixed cloudlet tasks and thus improve fog features. In this field there are already some early efforts.

Dongdong Ye et al. [25] show a use case where buses are used as mobile cloudlets. They leverage the characteristics of buses (e.g., the same routes, many stops) and propose a scalable fog computing paradigm with servicing offloading in bus networks. The bus fog servers not only provide fog computing services for the mobile users on bus, but also are motivated to accomplish the computation tasks offloaded by roadside cloudlets. By this way, the computing capability of roadside cloudlets is significantly extended.

[172] 2016 **Vehicular fog computing: A viewpoint of vehicles as the infrastructures.** Xueshi Hou et al. present the idea of utilizing vehicles as the infrastructures for communication and computation, named vehicular fog computing (VFC), which is an architecture that utilizes a collaborative multitude of end-user clients or near-user edge devices to carry out communication and computation, based on better utilization of individual communication and computational resources of each vehicle. They discussed four types of scenarios of moving and parked vehicles or congested traffic. Also, they point out the advantages against vehicular cloud computing (VCC) and the advantages in scenarios like of emergency operations for natural disaster and terrorist attack.

[186] 2018 **Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning.** Unmanned aerial vehicles (UAVs) have been considered as means to provide computing capabilities. In this model, UAVs act as fog nodes and provide

computing capabilities with enhanced coverage for IoT nodes. The system aims at minimizing the total mobile energy consumption while satisfying QoS requirements of the offloaded mobile application. This architecture is based on a UAV-mounted cloudlet which provides the offloading opportunities to multiple static mobile devices. They aim to minimize the mobile energy consumption, while satisfying QoS requirements and optimize UAV's trajectory.

[270] 2016 **An adaptive cloudlet placement method for mobile applications over gps big data.** Introduces the concept of movable cloudlets and explores the problem of how to cost-effectively deploy these movable cloudlets to enhance cloud services for dynamic context-aware mobile applications. To this end, Haolong Xiang et al. propose an adaptive cloudlet placement (via GPS) method for mobile applications. Specifically, the gathering regions of the mobile devices are identified based on position clustering, and the cloudlet destination locations are confirmed accordingly. Besides, the traces between the origin and destination locations of these mobile cloudlets are also achieved.

2.3 Handover

[26] The authors observe that traditional mobile network handover mechanisms cannot handle the demands of fog computation resources and the low-latency requirements of mobile IoT applications. The authors propose Follow Me Fog framework to guarantee service continuity and reduce latency during handovers. The key idea proposed is to continuously monitor the received signal strength of the fog nodes at the mobile IoT device, and to trigger pre-migration of computation jobs before disconnecting the IoT device from the existing fog node.

[27] Present a novel service handoff system which seamlessly migrates offloading services to the nearest edge server, while the mobile client is moving. Service handoff is achieved via container migration. They have identified an important performance problem during Docker container migration, proposing a migration method which leverages the layered storage system to reduce file system synchronization overhead, without dependence on the distributed file system.

[28] Develop a novel user-centric energy-aware mobility management (EMM) scheme, in order to optimize the delay, under energy consumption constraint of the user. Based on Lyapunov optimization and multi-armed bandit theories, EMM works in an online fashion. Theoretical analysis explicitly takes radio handover and computation migration cost into consideration and proves a bounded deviation on both the delay performance and energy consumption compared with the oracle solution with exact and complete future system information. The proposed algorithm also effectively handles the scenario in which candidate BSs randomly switch ON/OFF during the offloading process of a task.

[29] Study of mobility support issue in fog computing for guaranteeing service continuity. Propose a novel SDN enabled architecture that is able to facilitate mobility management in fog computing by decoupling mobility management and data forwarding functions. Design an efficient handover scheme by migrating mobility management and route optimization logic to the SDN controller. By employing link layer information, the SDN controller can

pre-compute the optimal path by estimating the performance gain of each path.

[30] To guarantee the strict latency requirements, new solutions are required to cope with the user mobility in a distributed edge cloud environment. The use of proactive replication mechanism seems promising to avoid QoE degradation during service migration between different edge nodes. However, accounting for the limited resources of edge micro data-centers, appropriate optimization solutions must be developed to reduce the cost of service deployment, while guaranteeing the desired QoE. In this paper, Ivan Farris et al., by leveraging on prediction schemes of user mobility patterns, have proposed two linear optimization solutions for replication-based service migration in cellular 5G networks: the min-RM approach aims at minimizing the QoE degradation during user handover; min-NSR approach favors the reduction of service replication cost. Simulation results proved the efficiency of each solution in achieving its design goal and provides useful information for network and service orchestrators in next-generation 5G cloud-based networks.

2.4 Data placement

[31] Enrique Saurez et al. propose Foglets, a programming model that facilitates distributed programming across fog nodes. Foglets provides APIs for spatio-temporal data abstraction for storing and retrieving application-generated data on the local nodes. Through the API, Foglets processes are set for a certain geospatial region and Foglets manages the application components on the Fog nodes. Foglets is implemented through container-based visualization. The API takes into account QoS and load balancing when migrating persistent (stateful) data between fog nodes. It provides various functionalities: automatically discovers fog computing resources deploys application components onto the fog computing resources commensurate with the latency requirements of each component in the application. It supports multi-application collocation on any compute node. Provides communication APIs for components that are deployed at different physical levels of the network hierarchy to communicate with one another to exchange application state. Lastly, it supports both latency- and workload-driven resource adaptation and state migration over space (geographic) and time to deal with the dynamism in situation awareness application.

[32] They propose a layered Fog framework to better support IoT applications through virtualization. The virtualization is divided into object virtualization (VOs), network function virtualization and service virtualization. VOs to address the protocol inconsistency (lack of unified networking protocols that leads to exaggerated overhead); Network function virtualization maps standard networking services to VOs, thus, minimize the communication process between consumers and producers by minimizing latency, improving security and scalability; Service virtualization that composes the community and Cloud Apps from various vendors to serve local Fog users with high quality of experience (QoE) but at low cost. At last, Foglets are involved to seamless aggregate multiple independent virtual instances, Fog network infrastructures, and software platforms.

[33] This paper proposes a Distributed Dataflow (DDF) programming model for the IoT that utilizes computing infrastructures across the Fog and the Cloud. Also, evaluate their proposal by implementing a DDF framework based on Node-RED (Distributed Node-RED or D-NR), a visual programming tool that uses a flow-based model for building IoT applications. To address challenges of the intrinsic nature of the IoT (heterogeneous de-

vices/resources, a tightly coupled perception-action cycle and widely distributed devices and processing), they propose a Distributed Dataflow (DDF) programming model for the IoT that utilities computing infrastructures across the Fog and the Cloud. Also, they evaluate their proposal by implementing a DDF framework based on Node-RED (Distributed Node-RED or D-NR), a visual programming tool that uses a flow-based model for building IoT applications.

[34] The authors address the problem of multi-component application placement on fog nodes. Each application could be modeled as a graph, where each node is a component of the application, and the edges indicate the communication between them.

[35] With the state-of-the-art virtualization technologies, services can be implemented in modular software as a graph/chain of portable VOs that can be dynamically migrated around the Telco infrastructure. It is proposed a VO clustering and migration policy that jointly considers user proximity and inter-VO affinity to scalably support user mobility, while allowing service differentiation among users.

2.5 Migration Optimization

As aforementioned, in section 2.2, mobility in fog computing is supported through VM migration. Therefore, considering an environment where both users and cloudlets are mobile, the decision-making of where to send the VMs, so that users' applications have their requirements fulfilled, is a major concern is such a dynamic environment. In the section 2.4 it was verified that applications cloud be seen as a whole, or as set of modules that may have different requirements. Regardless of their type, whenever justifies, it is necessary to re-adapt the whole system, where this is performed through the exchange of applications or modules between cloudlets, making use of VMs for this purpose. With this, it is necessary to answer the two questions: *When is this exchange justified? And how is it made?*

[36] Most work studying the placement of operators in such an environment completely disregards the migration costs. However, the mobility of users requires frequent migration of operators, together with possibly large state information, to meet latency restrictions and save bandwidth in the infrastructure. In this papers, Beate Ottenwälder et al. present a placement and migration method for providers of infrastructures that incorporate cloud and fog resources. It ensures application-defined end-to-end latency restrictions and reduces the network utilization by planning the migration ahead of time using predicted mobility patterns. Furthermore, present how the application knowledge of the complex event processing (CEP) system can be used to improve current live migration techniques for Virtual Machines (VMs) to reduce the required bandwidth during the migration. First, it allows us to amortize the migration costs by selecting migration targets that ensure a low expected network utilization for a sufficiently long time. Second, it allows us to serialize the operator for the migration and migrating parts of the operator a priori in away where unnecessary events are not migrated and bandwidth is reduced.

[37] This paper, Rahul Urgaonkar et al. present a model to optimize operational costs while providing rigorous performance guarantees as a sequential decision-making Markov Decision Problem (MDP). This model is different from the traditional solution methods (such as dynamic programming) that require extensive statistical knowledge and are computationally prohibitive. First they establish a decoupling property of the MDP that reduces it to two independent MDPs. Then, using the technique of Lyapunov optimization over renewals they design an online control algorithm that is provably cost-optimal. When the decoupling property holds, it enables the design of simple online control algorithms that do not require any knowledge of the underlying statistics of the MDPs, yet are provably optimal. This technique was applied to dynamic service migration and workload scheduling.

[38] Wuyang Zhang et al. propose SEGUE, a service that achieves optimal migration decisions by providing a long-term optimal QoS to mobile users. This model arises to overcome the limitations of previous studies that propose a static distance-based Markov Decision Process (MDP) for optimizing migration decisions that although works, it fails to consider dynamic network and server states. SEGUE is a MDP-based model which incorporates the two dominant factors in making migration decisions: 1) network state, and 2) server state. On top of that SEGUE answers the question of when to recalculate the MDP model, because to short intervals would create heavy overhead, and long intervals may translate into lazy migration. SEGUE adopts a QoS aware scheme to activate the MDP model when a QoS violation is predicted to solve for the when to migrate variable. Two components of SEGUE work together to achieve this. One module monitors network states, server workloads and

user mobility and the other is responsible for QoS prediction. This allows SEGUE to avoid unnecessary migration costs and bypass any possible QoS violations.

[39] Alberto Ceselli et al. present a link-path formulations supported by heuristics to compute solutions in reasonable time to firstly determining where to install cloudlet facilities and secondly assigning sets of access points, such as base stations to cloudlets and lastly supporting VM orchestration and considering partial user mobility information, as well as the satisfaction of service-level agreements. Qualify the advantage in considering mobility for both users and VMs. Compare two VM mobility modes, determining that high preference should be given to live migration and bulk migration seem to be a feasible alternative on delay-stringent tiny-disk services, such as augmented reality support, and only with further relaxation on network constraints. Also, they focus on the potential medium-term planning of an edge cloud network in mobile access networks. They study two design cases: 1) network in a static state 2) network state variations in terms of load and service level, caused by user mobility.

[40] User Equipment (UE) are moving in the network, and so the E2E (between UE and its Avatar) may become worse, degrading QoS. The live Avatar migration is triggered to adjust the location of the UE's Avatar. However, the **migration process consumes extra resources** of the Avatar that may degrade the performance of applications running in the Avatar. By considering the **gain (i.e., the end-to-end delay reduction)** and the **cost (i.e., the migration overheads)** of the live Avatar (a software clone; high performance Virtual Machine (VM) located in a cloudlet) migration, Xiang Sun et al. propose a PRofIt Maximization Avatar pLacement **(PRIMAL) strategy for the cloudlet network in order to optimize the trade-off between the migration gain and the migration cost by selectively migrating the Avatars to their optimal locations.

[41] Abdelwahab et al. design FogMQ, a self-deploying brokering clones that discover cloud/edge hosting platforms and autonomously migrate clones between them according to self-measured weighted tail end-to-end latency without the need of a central monitoring and control unit, not having to sacrifice computation offloading gain in cloud platforms. Finally, it is simple and requires no change in existing cloud platforms controllers.

[42] One problem remains unsolved: how to distribute the work among the user device, the edge clouds, and the center cloud to meet all three requirements especially when users are mobile. Wuyang Zhang et al. propose a hybrid gaming architecture that achieves clever work distribution. It places local view change updates on edge clouds for immediate responses, frame rendering on edge clouds for high bandwidth, and global game state updates on the center cloud for user scalability. In addition, they propose an efficient service placement algorithm based on a Markov decision process. This algorithm dynamically places a user's gaming service on edge clouds while the user moves through different access points taking into account the presence of dynamic network states and server workload states, and user mobility. However, unlike many of the service migration solutions which assumes an ignorable service transition time, they acknowledge that it is impossible to migrate an edge service from one edge to another instantly given the size of a VR game world. Therefore, they propose a mechanism to ensure a new edge cloud is activated when a player connects to the new one. It also co-places multiple users to facilitate game world sharing and reduce the overall migration overhead. Also, they derive optimal solutions and devise efficient

heuristic approaches and study different algorithm implementations to speed up the runtime.

[25] In this paper, they leverage the characteristics of buses and propose a scalable fog computing paradigm with servicing offloading in bus networks. The bus fog servers not only provide fog computing services for the mobile users on bus, but also are motivated to accomplish the computation tasks offloaded by roadside cloudlets. By this way, the computing capability of roadside cloudlets is significantly extended. They consider an allocation strategy using genetic algorithm (GA). With this strategy, the roadside cloudlets spend the least cost to offload their computation tasks. Meanwhile, the user experience of mobile users are maintained.

2.6 Multi-objective

QoS, QoE, Cost, Energy, Handover, Mobility, Bandwidth

2.7 Toolkits

[43] In this paper they propose a simulator, called iFogSim, to model IoT and Fog environments and measure the impact of resource management techniques in latency, network congestion, energy consumption, and cost.

[24] An extension of iFogSim to support mobility through migration of VMs between cloudlets.

[44] The authors propose another edge computing simulation environment, EdgeCloudSim, that considers both network and computational resources and covers all aspects of edge computing simulation ,including network and computational modelling. Similar to iFogSim, EdgeCloudSim relies on CloudSim as well. Additionally, EdgeCloudSim provides a modular architecture to provide support for a variety of critical functionality and supports simulating multi-tier scenarios where multiple edge servers are running in coordination with upper layer cloud solutions.

3 Architecture

To this work is considered the typical architecture of fog computing, Fig. 2, where apart from the fixed cloudlets, there are some mobile fog nodes that can be used to support individual applications or to support the fixes cloudlets, providing offloading support ..... tal como o paper do bus

4 Evaluation

The evaluation of the proposed architecture will be done xxxx

5 Schedule of Future Work

Future work is scheduled as follows:

- xxxx
- xxxx

6 Conclusion

xxxxx

References

1. "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021 white paper - cisco," https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper- c11-520862.html, (Accessed on 10/25/2018).
2. S. R. Ellis, K. Mania, B. D. Adelstein, and M. I. Hill, "Generalizeability of latency detection in a variety of virtual environments," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 48, no. 23. SAGE Publications Sage CA: Los Angeles, CA, 2004, pp. 2632–2636.
3. B. Taylor, Y. Abe, A. Dey, M. Satyanarayanan, D. Siewiorek, and A. Smailagic, "Virtual machines for remote computing: Measuring the user experience," *Carnegie Mellon University*, 2015.
4. T. Szigeti and C. Hattingh, *End-to-end qos network design*. Cisco press, 2005.
5. M. Satyanarayanan, "Cloudlets: at the leading edge of cloud-mobile convergence," in *Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures*. ACM, 2013, pp. 1–2.
6. F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.
7. S. Davy, J. Famaey, J. Serrat-Fernandez, J. L. Gorricho, A. Miron, M. Dramitinos, P. M. Neves, S. Latré, and E. Goshen, "Challenges to support edge-as-a-service," *IEEE Communications Magazine*, vol. 52, no. 1, pp. 132–139, 2014.
8. T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," *IEEE Network*, vol. 27, no. 5, pp. 12–19, 2013.
9. A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *arXiv preprint arXiv:1808.05283*, 2018.

10. M. Satyanarayanan, "Fundamental challenges in mobile computing," in *Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing*. Acm, 1996, pp. 1–7.
11. M. Shiraz, A. Gani, R. H. Khokhar, and R. Buyya, "A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1294–1313, 2013.
12. "Open edge computing," http://openedgecomputing.org/about.html, (Accessed on 10/24/2018).
13. P. by the OpenFog, C. Architecture, and W. Group, "Openfog reference architecture for fog computing," 0208.
14. "Etsi - multi-access edge computing," https://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing, (Accessed on 10/25/2018).
15. "Mobile edge computing vs. multi-access edge computing," https://www.sdxcentral.com/mec/definitions/mobile-edge-computing-vs-multi-access-edge-computing/, (Accessed on 10/25/2018).
16. T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.
17. K. P. Kadiyala and J. A. Cobb, "Inter-as traffic engineering with sdn," in *Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017 IEEE Conference on*. IEEE, 2017, pp. 1–7.
18. M. Satyanarayanan, V. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE pervasive Computing*, 2009.
19. Y. Jararweh, L. Tawalbeh, F. Ababneh, and F. Dosari, "Resource efficient mobile computing using cloudlet infrastructure," in *Mobile Ad-hoc and Sensor Networks (MSN), 2013 IEEE Ninth International Conference on*. IEEE, 2013, pp. 373–377.
20. "Cisco pushes iot analytics to the extreme edge with mist computing - rethink," https://rethinkresearch.biz/articles/cisco-pushes-iot-analytics-extreme-edge-mist-computing-2/, (Accessed on 10/25/2018).
21. J. S. Preden, K. Tammemäe, A. Jantsch, M. Leier, A. Riid, and E. Calis, "The benefits of self-awareness and attention in fog and mist computing," *Computer*, vol. 48, no. 7, pp. 37–45, 2015.
22. "What is fog computing? : Openfog consortium," https://www.openfogconsortium.org/page-section/definition-of-fog-computing/, (Accessed on 10/24/2018).
23. L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, "Mobility-aware application scheduling in fog computing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 26–35, 2017.
24. M. M. Lopes, W. A. Higashino, M. A. Capretz, and L. F. Bittencourt, "Myifogsim: A simulator for virtual machine migration in fog computing," in *Companion Proceedings of the10th International Conference on Utility and Cloud Computing*. ACM, 2017, pp. 47–52.
25. D. Ye, M. Wu, S. Tang, and R. Yu, "Scalable fog computing with service offloading in bus networks," in *Cyber Security and Cloud Computing (CSCloud), 2016 IEEE 3rd International Conference on*. IEEE, 2016, pp. 247–251.
26. W. Bao, D. Yuan, Z. Yang, S. Wang, W. Li, B. B. Zhou, and A. Y. Zomaya, "Follow me fog: Toward seamless handover timing schemes in a fog computing environment," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 72–78, 2017.
27. L. Ma, S. Yi, and Q. Li, "Efficient service handoff across edge servers via docker container migration," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. ACM, 2017, p. 11.
28. Y. Sun, S. Zhou, and J. Xu, "Emm: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2637–2646, 2017.
29. Y. Bi, G. Han, C. Lin, Q. Deng, L. Guo, and F. Li, "Mobility support for fog computing: An sdn approach," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 53–59, 2018.

30. I. Farris, T. Taleb, M. Bagaa, and H. Flick, "Optimizing service replication for mobile delay-sensitive applications in 5g edge network," in *Communications (ICC), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–6.
31. E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwälder, "Incremental deployment and migration of geo-distributed situation awareness applications in the fog," in *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems*. ACM, 2016, pp. 258–269.
32. J. Li, J. Jin, D. Yuan, and H. Zhang, "Virtual fog: A virtualization enabled fog computing framework for internet of things," *IEEE Internet Things J*, vol. 5, pp. 121–131, 2018.
33. N. K. Giang, M. Blackstock, R. Lea, and V. C. Leung, "Developing iot applications in the fog: a distributed dataflow approach," in *Internet of Things (IOT), 2015 5th International Conference on the*. IEEE, 2015, pp. 155–162.
34. T. Bahreini and D. Grosu, "Efficient placement of multi-component applications in edge computing systems," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. ACM, 2017, p. 5.
35. R. Bruschi, F. Davoli, P. Lago, and J. F. Pajo, "Move with me: Scalably keeping virtual objects close to users on the move," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.
36. B. Ottenwälder, B. Koldehofe, K. Rothermel, and U. Ramachandran, "Migcep: operator migration for mobility driven distributed complex event processing," in *Proceedings of the 7th ACM international conference on Distributed event-based systems*. ACM, 2013, pp. 183–194.
37. R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, 2015.
38. W. Zhang, Y. Hu, Y. Zhang, and D. Raychaudhuri, "Segue: Quality of service aware edge cloud service migration," in *Cloud Computing Technology and Science (CloudCom), 2016 IEEE International Conference on*. IEEE, 2016, pp. 344–351.
39. A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Transactions on Networking (TON)*, vol. 25, no. 3, pp. 1818–1831, 2017.
40. X. Sun and N. Ansari, "Primal: Profit maximization avatar placement for mobile edge computing," in *Communications (ICC), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.
41. S. Abdelwahab, S. Zhang, A. Greenacre, K. Ovesen, K. Bergman, and B. Hamdaoui, "When clones flock near the fog," *IEEE Internet of Things Journal*, 2018.
42. W. Zhang, J. Chen, Y. Zhang, and D. Raychaudhuri, "Towards efficient edge cloud augmentation for virtual reality mmogs," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. ACM, 2017, p. 8.
43. H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. 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*, vol. 47, no. 9, pp. 1275–1296, 2017.
44. C. Sonmez, A. Ozgovde, and C. Ersoy, "Edgecloudsim: An environment for performance evaluation of edge computing systems," in *Fog and Mobile Edge Computing (FMEC), 2017 Second International Conference on*. IEEE, 2017, pp. 39–44.