Accepted Manuscript

Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities

Mohammad Aazam, Sherali Zeadally, Khaled A. Harras

PII: S0167-739X(18)30197-3
DOI: https://doi.org/10.1016/j.future.2018.04.057
Reference: FUTURE 4135

To appear in: *Future Generation Computer Systems*

Received date : 19 February 2018
Revised date : 12 April 2018
Accepted date : 19 April 2018

Please cite this article as: M. Aazam, S. Zeadally, K.A. Harras, Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities, *Future Generation Computer Systems* (2018), https://doi.org/10.1016/j.future.2018.04.057

Offloading in Fog Computing for IoT: Review, Enabling Technologies, and Research Opportunities

Mohammad Aazam[a], Sherali Zeadally[b], Khaled A. Harras[a]

[a] *Carnegie Mellon University, Qatar*
aazam@ieee.org
kharras@cs.cmu.edu
[b] *University of Kentucky, USA*
szeadally@uky.edu

Abstract

The digital world is expanding rapidly and advances in networking technologies such as 4G long-term evolution (LTE), wireless broadband (WiBro), low-power wide area networks (LPWAN), 5G, LiFi, and so on, all of which are paving the way for the emergence of sophisticated services. The number of online applications is increasing along with more computation, communication, and intelligent capabilities. Although current devices in use today are also getting more powerful in terms of features and capabilities, but they are still incapable of executing smart, autonomous, and intelligent tasks such as those often required for smart healthcare, ambient assisted living (AAL), virtual reality, augmented reality, intelligent vehicular communication, as well as in many services related to smart cities, Internet of Things (IoT), Tactile Internet, Internet of Vehicles (IoV), and so on. For many of these applications, we need another entity to execute tasks on behalf of the user's device and return the results - a technique often called offloading, where tasks are outsourced and the involved entities work in tandem to achieve the ultimate goal of the application. Task offloading is attractive for emerging IoT and cloud computing applications. It can occur between IoT nodes, sensors, edge devices, or fog nodes. Offloading can be performed based on different factors that include computational requirements of an application, load balancing, energy management, latency management, and so on. We present a taxonomy of recent offloading schemes that have been proposed for domains such as fog, cloud computing, and IoT. We also discuss the middleware technologies that enable offloading in a cloud-IoT cases and the factors that are important for offloading in a particular scenario. We also present research

opportunities concerning offloading in fog and edge computing.

Keywords: Cloud computing, fog computing, edge computing, Internet of Things (IoT), middleware, offloading.

1. Introduction

Technology has become an essential part of human life and it keeps improving over time. Whether it be devices that can communicate, such as smartphones, tablet computers; communication technologies such as 4G, long-term evolution (LTE); or new concepts and paradigms such as cloud computing, Internet of Things (IoT), and semantic web, that enable innovative applications. It is only a matter of time after which IoT applications and devices will be making their way into every aspect of our lives with technologies in smart homes, smart cities, factories, agriculture, green energy, healthcare, food, emergency and disaster management, automotive, aerospace, and telecommunication [1, 2, 3, 4, 5]. These applications will face different challenges involving interoperability, scalability, usability, privacy, and security, to name a few. Cloud computing, on the other hand, has matured rapidly over the last decade. Several commercial products today provide a platform for on-demand access to various compute and network resources with greater capabilities in terms of storage and processing power, thereby enabling ubiquitous access to cloud services from anywhere and at any time. Hence, the integration of cloud and IoT, creating the cloud of things (CoT) will play an important role in the future Internet. Likewise, the cloud can also benefit from IoT in terms of managing real world services more dynamically and in a distributed way by leveraging IoT resources. As reported in [6], 90% of global Internet users are now dependent on cloud-based services, whether directly through consumer services or indirectly through their service providers dependence upon various commercial clouds.

Furthermore, mobile computing is also in widespread use today. The number of connected devices per person is going to be 6.58 on average by 2020 [6]. Current smart phones contain, on average, more than a dozen sensors. With various smart devices emerging on the market, a wide variety of mobile services have appeared lately. The services range from cloud storage, healthcare, IoT, instant messaging (IM), to augmented reality, multimedia streaming, navigation, and many others [7, 8, 9]. Managing such diverse service requests from billions of nomadic mobile users becomes extremely

difficult for the centralized cloud [6]. Mobile devices are generally resource-constrained and with their mobility attribute, often it becomes necessary to offload some of their tasks to the cloud. The rapid development of wearable devices such as Google Glass, Intel's Vaunt smart glasses, Gear smartwatches, and others is also increasing their reliance on the cloud computing environment. Given the large distance between the cloud and end-user devices, it becomes a challenge to support real-time processing and deliver fast response times to end-users. To address this challenge, some middleware is required between the end nodes and the cloud. Various middleware technologies such as mobile edge computing, cloudlets, and so on, that somehow come under the umbrella of fog computing have been proposed and discussed in the literature [10, 11, 12, 13, 14, 15, 16].

Additionally, many services (such as online gaming, image processing) now require artificial intelligence (AI) to be incorporated into them in order to achieve the required smartness, intelligence, and autonomy. As applications become more sophisticated and intelligent, we need to efficiently manage the execution of increasingly complex tasks based on the requirements of the application. Some of these tasks' resource requirements could be well beyond the capabilities of the end-user's device. In these cases, the tasks are offloaded to the middleware which may in turn offload them to the cloud. However, it is worth noting that tasks are not bound to computation only but also to other resources such as storage [17].

The main contributions of this work are:

- We present the various criteria used by recently proposed middleware technologies when tasks are offloaded in the fog computing environment.
- We describe the key technologies that are currently being used to enable offloading in fog computing and we describe some common scenarios where offloading may occur.
- We identify future research challenges that still need to be addressed to improve tasks' offloading performance, efficiency, and reliability in fog computing.

2. Criteria Used in Offloading

In this section, we discuss some of the criteria that are used when deciding whether or not to offload certain tasks. Figure 1 depicts a generalized

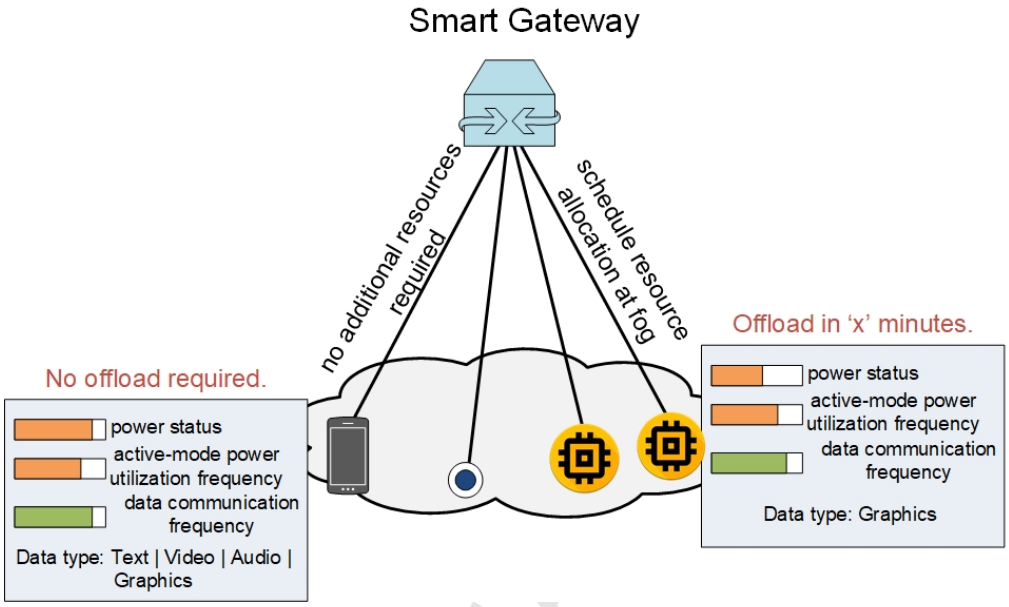

Figure 1: The smart gateway decides if an underlying node requires task offloading.

decision-making scenario, regarding whether or not to offload tasks. A middleware, such as a smart gateway, monitors the underlying nodes and decides if offloading is needed or not.

2.1. Excessive computation or resource constraint

When an application requires computation more than the capability of the native device, certain tasks must be offloaded to a comparatively resource-rich device [18]. For example, a geographical map service that executes on a smartphone requires a cloud to update the map through satellite data.

2.2. To meet latency requirement

Distance can affect delay-sensitive applications. In this case, a node close to the proximity of the receiving node must be involved to offload the task from the distant node and provide the required services with minimum delays [19]. For example, a cloud hosting multimedia services may have to offload a part or the entire content needs to be cached at an edge device or fog server residing close to the proximity of the user.

2.3. Load balancing

When one server has reached its capacity of executing tasks, additional tasks will need to be distributed among other servers within the service provider's ecosystem [20]. For example, a cloud datacenter manages tasks by appropriately distributing them among different servers in the datacenter. Similarly, a fog micro datacenter having multiple servers would distribute tasks to balance the load of the incoming requests.

2.4. Permanent or long-term storage

Long-term storage requires a lot of space, depending upon the type of the service. It is not feasible to satisfy those services that require a large amount of storage on small devices such as smartphones, fog servers, edge nodes, and so on. In such cases, storage space is reserved and can be allocated at a resource-rich node such as a cloud server.

2.5. Data management and organization

When data is moved from one device to another, one of the reasons may be to organize the data. Rarely used data may not be required anymore at the native device, but at the same time, the data may be needed in the future and therefore needs to be saved somehow. In such a case, data has to be offloaded to another secure location such as from a smartphone to a cloud or some private cloud [21], [17].

2.6. Privacy and security

Depending on the sensitivity and secrecy of the data or tasks, offloading may take place because of privacy and security. For example, an enterprise or hospital's data or tasks may be moved from local machines to a private cloud. Similarly, personal data from a smartphone may be moved to a personal mobile edge cloud [21].

2.7. Accessibility

If the native device from where the task or data was created is not accessible from everywhere, the task or tasks can be offloaded to a cloud-like node where the data can be accessed from anywhere, anytime.

2.8. Affordability, feasibility, and maintenance

In many cases, it would not be feasible to maintain high-end servers for executing occasional compute-extensive or storage related tasks. In such cases, tasks have to be offloaded to the service providers that provide pay-as-you-go services, such as those provided by a cloud. This might be more appropriate because the hassle of maintaining devices and the need for maintenance staff can be avoided.

3. IoT Middleware Technologies

Middleware technologies play an important role in IoT applications, especially when it comes to reduce energy consumption and latency. In this section, we present basic middleware technologies that can be useful in various mobile and IoT service provisioning architectures. Such middleware technologies, but not limited to them, come under the concept of fog or edge computing paradigm.

3.1. Cloudlet

Cloudlets [10], [22] provide computing and storage resources to nearby mobile devices. They are highly decentralized and dispersed infrastructures. They have been proposed to overcome some of the issues such as wide-area network (WAN) latency, jitter, and packet loss faced by the mobile cloud computing architecture. They have a very rich pool of resources and have access to high-speed communication links for fast access to the services in the cloud [10]. Cloudlet brings cloud closer to mobile devices and user can offload their tasks to these cloudlets to save battery and other resources, such as storage, computation, and memory. Unlike clouds, cloudlets are usually much closer to mobile users, typically one hop away and are accessible through high speed wireless connections using mobile broadband, WiFi, and so on. The advantages of cloudlets include: lower latency, higher bandwidth, offline availability, and cost-effectiveness [16].

3.2. Mobile edge computing

Mobile edge computing (MEC) was proposed in 2014 by the European Telecommunications Standard Institute (ETSI) for the first time. ETSI defined MEC as a resource that provides IT and cloud-computing capabilities within the radio access network (RAN) and is physically close to mobile [23]. MEC has three main components: edge devices that include mobiles and

IoT devices, edge cloud, and public cloud. Edge devices are connected to the network. The edge cloud is deployed in the base station and has less resources as compared to the public cloud which can be accessed through the Internet. Furthermore, the edge cloud controls the network traffic and hosts different mobile edge applications [24]. MEC enables mobile users to access IT and cloud services in their vicinity within the RAN range. It aims to reduce end-to-end latency by bringing the processing and storage services to the edge of the network. It helps to support context-aware services to end-users by providing real-time RAN information, which is used by application and content developers. As a result, it improves quality-of-experience (QoE) and user-satisfaction. It increases the responsibilities of the edge and allows it to host services which result in: (a) reduced latency over the network and (b) low consumption of network bandwidth. New applications and services can be rapidly deployed to users (mobile subscribers, enterprises) by allowing third party partners to handle the radio network edge [24].

3.3. Micro datacenter

Micro datacenter (MDC) is a smaller datacenter architecture that is portable and containerized [25][26]. MDC is usually a stack of 4 or less servers per rack. MDC is a modular datacenter, where capacity can be increased or decreased according to the type of application. MDC comes with the features a traditional datacenter may have, such as cooling system, security, and fire-protection, but on a smaller scale so that it can be deployed indoors and outdoors including in tough terrains.

3.4. Nano datacenter

Nano datacenters (NaDa or nDCs) are always-on gateways that have additional capabilities in terms of memory and computation. Multiple applications, such as video-on-demand (VoD) and gaming can take benefit from NaDa [27]. The noteworthy benefits of NaDa are that they are cheaper to deploy, reduce traffic load and variability on the backbone, are highly scalable for the ISPs and flexible for the users. They can also provide localized and personalized services. However, their uplink bandwidth is often limited.

3.5. Delay tolerant network

Delay tolerant network (DTN), compared to traditional network paradigm, is a mobile ad-hoc network (MANET) suitable for situations where end-to-end connectivity from the source mobile to the destination mobile is not

possible. The main goal of the DTN architecture is to handle problems such as long delays and communication interruptions prevalent in long distance communications [28]. It works on the principle of store-carry and forward [29]. The mobile nodes between the source and the destination keep the data with them until they find a suitable mobile node in the destination path and forward the data to that node [30]. The connectivity is intermittent because of the sparse topology. Disruptions in the connections are very frequent and delays are generally long [29].

3.6. Femto cloud

A femto cloud is an orchestration of co-located devices in order to harvest computational or storage capabilities [11]. Most of the times, mobile devices are under-utilized. Offloading tasks to nearby devices in areas such as in a public transit, classroom, a coffee shop, and so on, can be cost-effective and performance efficient.

Table 1 provides the characteristics of the discussed IoT middleware technologies.

Table 1: Characteristics of IoT middleware technologies.

Middleware	Target Nodes	Distance	Core Purpose	Scalability
Cloudlet	mobile	1-hop	latency management	medium to high
Mobile edge computing	mobile and IoT	1-hop	latency management	medium to high
Micro datacenter	mobile and IoT	1 to 2 hops	latency management, task offloading	high
Nano datacenter	mobile and IoT	1 to 2 hops	latency management, task offloading	low

Continued on next page

Table 1 – continued from previous page

Middleware	Target Nodes	Distance	Core Purpose	Scalability
Delay tolerant networking	mobile and IoT	1-hop	data delivery	low
Femto cloud	mobile and IoT	0-hop	task offloading	low

4. Enabling Technologies for Offloading Tasks in Fog Computing

Recent advances in communication technologies have opened up opportunities for offloading certain tasks to nearby systems which implement different types of middleware technologies. We present a brief review of some of the common networking technologies that are being used for offloading tasks below.

4.1. Wireless technology

IEEE 802.11 Wi-Fi is the most common form of wireless Internet access, with its latest standard IEEE 802.11ac, connectivity speed is faster than Gigabit Ethernet. Similarly, on-the-go Internet access technologies, such as wireless broadband (WiBro), mobile broadband, long-term evolution (LTE) provide connectivity to mobile nodes such as smartphones, tablet computers, vehicles, drones, and so on. Short-range technologies, such as Wi-Fi Direct, Bluetooth, and so on, are also worth mentioning here in the scenario where an industrial sensor network, a smart home network, or a healthcare body area network is offloading tasks to a nearby edge or fog device. All these networking technologies have paved the way for high-speed communications. For example, currently, 4G LTE can provide up to 100Mbps of download speed. 5G technologies already promise data rates up to 10Gbps. Higher data rates would enable faster offloading of tasks.

4.2. Smart, intelligent, and autonomous applications

With AI, machine learning, and context-aware services, applications would be able to decide on their own when and where to offload. More and more autonomy would take place, as a result of which, offloading of tasks would be

much easier and seamless. One of the examples of such autonomous applications is given here in detail [17]. Samsung Smart Switch is another example of smart applications that seamlessly transfers contacts, photos, calendar, and other items from one Samsung device to another.

4.3. Virtualization and containment

Virtualization (with multiple virtual machines (VMs) can be created over a single physical machine) has been extensively deployed in recent years to logically distribute and manage resources. For instance, multiple virtual sensors can be created over a physical sensor through virtualization as described in [31] [32]. Virtualization not only provides cost efficiency but also portability, where a whole virtual machine, or a virtual sensor can be offloaded or migrated to another node. Container technology is another promising technology where a container is an alternative to the VM in a sense that a container allows virtualization of the operating system. In contrast, VM virtualizes the physical machine by hosting multiple operating systems. The container software includes the runtime system tools, code, system libraries, and everything required to execute some software application. Being lightweight, a container is portable and hence, can play an important role in terms of live migration during task offloading.

4.4. Parallelism

Tasks can be very large and may require execution in parallel at different nodes. With parallelism available through multi-core technology running jobs simultaneously would be convenient and would facilitate task offloading in a fog computing environment.

5. Task Offloading in Fog Computing Application Scenarios

Offloading can take place at different locations, depending upon the type of service. The location where to offload is significant because it determines what kind of algorithms need to be executed and the tradeoffs that must be considered. These tradeoffs include: how fault (such as in communication, bandwidth aggregation, scheduling) is to be monitored and the kind of incentives (for executing offloaded tasks) that are to be paid, depending on the type of the device. In this section, we present some of the most popular scenarios for task offloading in a fog computing environment. Figure 2 presents an overview of some of scenarios where offloading can be executed.

The figure shows different locations of offloading. For example, communication call A represents direct access to the cloud, such as in the case of a smartphone accessing a cloud service. B represents offload to fog, where data may be aggregated or fog is assisting the cloud in providing any latency-sensitive or security-sensitive service. C represents IoT nodes accessing the smart gateway which acts as a middleware, evaluates the request and decides on whether to offload to the cloud instead of offloading to the fog depending on the requirements of the service. D represents a fog offloading to the cloud or the cloud requesting some data from the fog.

5.1. IoTs, sensors, and devices offloading to fog

5.1.1. Healthcare sensors offloading to fog

A wireless body area network (WBAN) gathering vital signs and remote monitoring data from a patient would require to offload diagnostic related tasks to an associated fog server. Fog applies data analytics and machine learning based techniques on the acquired data and sends the results back to the application running on the patients side.

5.1.2. Roadside units offloading to fog

In an intelligent transportation system (ITS), roadside units (RSUs), equipped with fog units - such as fog server or nDC, can assist the drivers and vehicles in many ways, such as real-time smart speed signs, smart signals according to traffic load and situation (e.g. emergency), assisting in-vehicle entertainment applications, and so on. On their own, RSUs are not rich in computational resources and hence, they require fog computing resources for more intensive or complex processing. For example, real-time traffic monitoring and management [33], smart traffic signal control [34] according to the traffic load and emergency situation [35], and so on.

5.1.3. Industrial sensors, actuators, and robots offloading to fog

In an industrial environment, sensors, actuators, and robots require a micro datacenter fog to manage the control of machines. In particular, in a manufacturing environment where large machines are being operated, tasks are often complex and many of them require extreme precision, accuracy with minimal delays. For such an environment, a middleware such as a localized and enriched fog micro datacenter can assist smart factory applications.

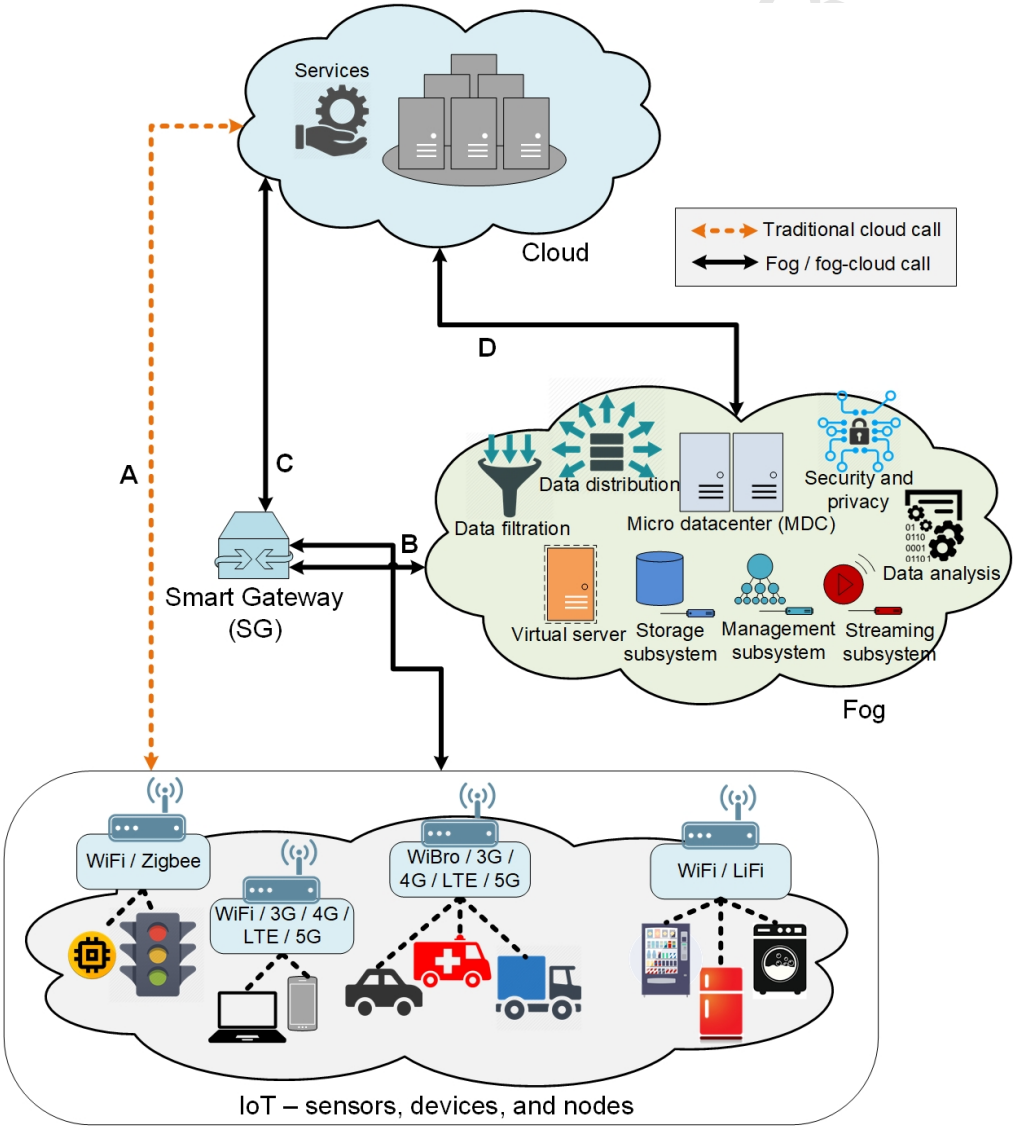

Figure 2: Smart gateway as a middleware to decide where to offload a certain task.

5.2. IoT/devices offloading to edge nodes

5.2.1. Smart glasses offloading to smartphone

Gadgets and smart devices have become an essential part of our daily lives. For instance, a device that offloads to an edge device can be a Google Glass capturing audio in a foreign language and sending to the smartphone where a client software translates the audio. Similarly, various visual data (e.g. from Intel Vaunt smart glasses) can be offloaded to the smartphone for image recognition or akin purposes [36].

5.2.2. Smartwatches offloading to smartphone

Extending the above example, a smartwatch (e.g. Gear) collects data about some physical activity that can include pulse and heartbeat and transmits it to the smartphone, where an application can process the data and make suggestions to the patient/user on how to adjust his/her activities and lifestyle according to his or her health conditions.

5.2.3. Smart home devices offloading to edge nodes

In a smart home, where, say, a vacuum cleaner may operate autonomously and would be controlled by an edge device that guides the vacuum cleaner if a piece of paper or trash is left behind. In this case, the vacuum cleaner sensor offloads image recognition tasks to the edge device.

5.3. Fog and edge nodes offloading to cloud

A fog or edge node would be offloading tasks to a cloud in various cases. The most straightforward or reasonable case would be when long-term storage is required (e.g., a daily data log of a patients healthcare system at the end of the day, or drivers behavior monitoring log at the end of the trip) or when the task is delay tolerant (such as a smartphone requesting satellite map data from the cloud), and so on.

5.4. Cloud offloading to fog

A cloud actually offloads tasks to a fog when it realizes that the latency requirements of the application cannot be achieved by the cloud computing environment. In this case, the cloud requires a local fog to provide service on its behalf. In this case, data is cached at the fog that is located close to the customer. Additionally, other situations where a cloud would offload processing to a fog is when the cloud requires more precise location information, such as in the case of drone operations; or exact location coordinates and speed

of fast-moving vehicles such as in a vehicular ad hoc network (VANET) [37]. Similarly, a cloud may offload tasks to a fog when the application requires low communication overheads over the core network. Hence, the communication is kept within a certain network domain and only the necessary data is communicated over the core network to the cloud, for example a visual sensor network, where CCTV cameras are constantly uploading video content to the central cloud.

5.5. Distributed offloading between fog and cloud

In the previous sections (5.3 and 5.4), we elaborate the cases where a fog node offloads to a cloud or vice versa, and the rest of the service is provided by the offloadee. However, there is another scenario where the service is still being provisioned and the offloading of tasks continues between a fog and a cloud while the service is being rendered. Such a case can occur in the scenario where both entities, the fog and the cloud, require some particular kind of resource or data from each other. That is, both the fog and the cloud work together in a distributive way to fulfill an application's needs. Nevertheless, this scenario can occur when the above two cases mentioned in Sections 5.3 and 5.4 are used simultaneously. For instance, a fog provides real-time control and maneuver of a group of drones performing monitoring tasks during the snow season in Canada, or forest-fire in Australia, may require the cloud to provide input based on previous seasons' historical data, so that the fog can adapt and provide more accurate results and vice versa. In this way, the fog and the cloud distribute the tasks to each other, and provide the service in tandem.

5.6. Cloud offloading to IoT/devices/sensors

Although not very common, but cloud offloading directly to IoT devices and sensors can occur when a cloud (even being rich in resources) requires some task to be performed by underlying sensors or devices so that the cloud can use the results returned to provide some service. For instance, a cloud providing virtual sensor networking [38] [39] would require offloading tasks to real, physical sensors in order to retrieve the actual data.

5.7. Fog offloading to another fog and cloud to another cloud

Offloading between servers or datacenters can take place mainly to achieve load balancing. When a server, whether it is fog server or a cloud server, has run out of resources, it offloads the additional tasks to another server within

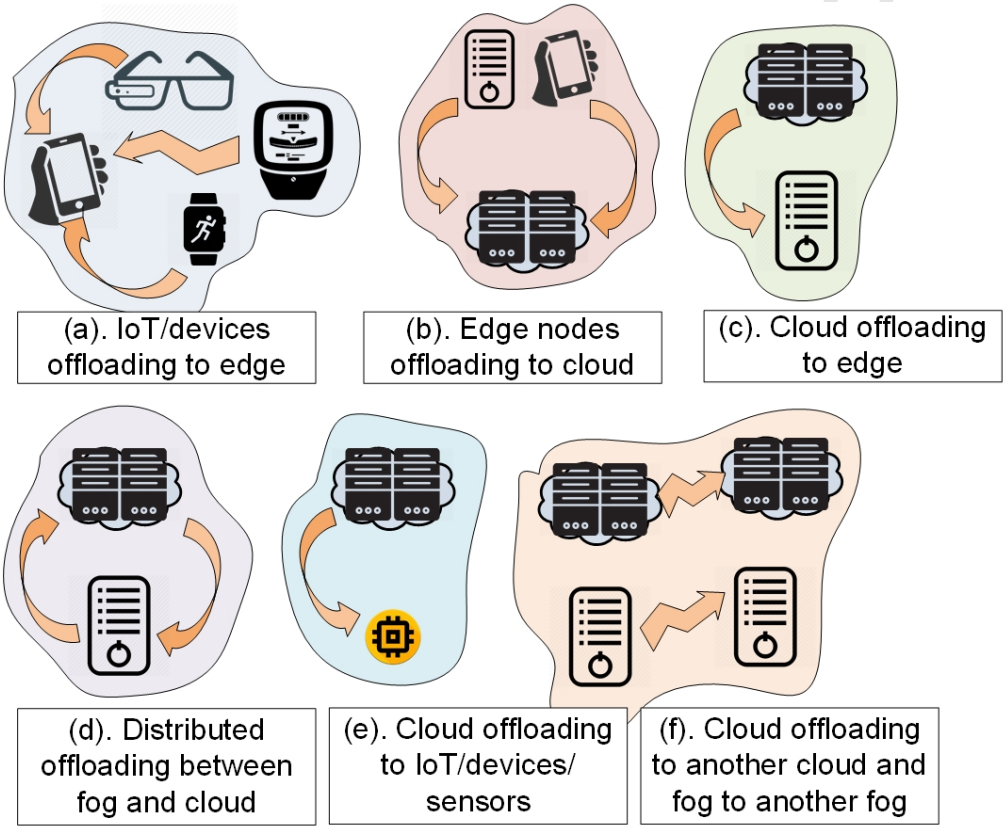

Figure 3: Offloading in fog computing scenarios.

the datacenter. In this way, inter-fog and inter-cloud communication takes place to better handle further incoming requests. Another reason for such offloading could be because of accessibility. For example, in an IoV scenario [40] where in-vehicle entertainment is being provided, one fog caches a certain amount of data for the vehicles accessing that service. As the vehicles move, the rest of the data is cached at the fog or cloud that is supposed to handle the approaching vehicles. This can be achieved if the fogs or the clouds are in sync with each other and the tasks are properly distributed.

Figure 3 shows offloading locations, where in case of (d), the offloading and service rendering take place in tandem. In rest of the cases ((a), (b), (c), (e), and (f)), the remaining part of the service post-offloading is provisioned by the offloadee.

6. Proposed Offloading Techniques in Fog Computing

In this section, we review some of the recently proposed offloading techniques in fog computing based on different factors.

Zhao et al. [41] discussed mobile device computational offloading. The authors stated that demands for mobile devices to run application requiring high computation is increasing. Compared to larger devices, such as a desktop computer, mobile phones still have limited computation resources mainly due to the limitation of its physical size. As a result, energy becomes limited as well, which means computation has to be offloaded to nearby resources that can fulfill the tasks. The mobile device can offload tasks to the cloud but it may not be feasible in the case where delay is intolerable. Therefore, fog computing becomes a good choice. However, this leads to another concern that the fog has less resources as compared with the cloud. Hence, energy consumption is one of the key parameters to be considered, while deciding about offloading tasks. In this context, the authors proposed an offloading algorithm that aims at minimizing the energy consumption at fog. First, they compute energy of the processing will consumer at fog nodes and cloud nodes. Based on the result obtained, the mobile device makes a choice whether to offload the computation to the cloud or fog after evaluating the energy consumption required by the fog and the cloud. Overall, the proposed scheme works on the basis of end to end delay (that includes transmission time and getting the results back) and energy consumption at the fog. The authors applied their scheme to a scenario where a mobile device is communicating with a fog. The fog can execute the required tasks on its own or forward them to the cloud. If the fog executes the tasks, the delay will comprise of the transmission delay of the input data to the fog together with the execution time at fog which will depend on the fog's processing capacity. If the task requires offloading to the cloud, the delay would be same but will also include the delay of communication from the fog to the cloud. This is because the authors assume that the fog is working as a mediator and every task first traverses the fog that then decides if the requested task should be executed locally at the fog itself, or at the cloud. In the case of energy consumption in the fog-only scenario, the total energy consumption is the idle energy consumption of mobile device when the fog server executes the instructions along with the energy consumption for the mobile device necessary to transfer the input data to the fog. If the cloud is also involved, the additional energy consumption would include the energy spent by the

fog in delegating the tasks to the cloud and the energy spent by the cloud executing them.

In the case of energy consumption in the fog-only scenario, the total energy consumption would be comprised of the idle energy consumption of mobile device when the fog server executes the instructions, plus, the energy consumption for the mobile device necessary to transfer the input bits to the fog. In case the cloud is also involved, the additional energy consumption would be that of fog transmitting the tasks to the cloud and the cloud then executing them.

Liang et al. [42] proposed a software defined networks (SDN) based fog offloading architecture. The authors argued that offloading of tasks should be performed through different offloading policies that are set according to the applications requirements. In this context, SDN can be very helpful because it enables the customization of policies for different flows.

Fricker et al. [43] used load balancing to trigger offloading the tasks within fog datacenters. Since fog computing has a smaller capacity compared to the traditional clouds, it is subject to saturation much faster. The authors described the scenario where a request arrives at an overloaded datacenter, and gets forwarded to neighboring datacenter with the same probability of receiving request. Datacenters have a large number of servers and if a lot of traffic causes congestion at one of the servers, other datacenters or servers may help mitigate the congestion by accepting some of the requests rejected or blocked at the congested servers or datacenter. In this way, the overall service response can be improved. In their approach, further requests are rejected/blocked based on whether it can offload tasks once a datacenter becomes overloaded. Essentially, the authors used a request blocking rate based offloading of tasks.

Hasan et al. [44] discussed incentive-based cloud-IoT offloading schemes for mobile nodes. The authors highlighted that the propagation of IoT devices has caused a paradigm shift in communication and computing. The authors proposed a mechanism called Aura, through which mobile client can create an ad hoc cloud of IoT and other computing devices within their physical environment. In this way, local computing resources from underloaded devices can be harvested and to execute offloaded tasks from mobile devices. IoT devices such as appliances, smartphones, intelligent vehicles, smart thermostats, and so on, all have computing capabilities. Although each IoT device has some specific purpose, however, it is common that such devices remain idle for a reasonable amount of time. During this idle period,

the computation capability can be outsourced to other nearby needy nodes through offloading. To motivate idle devices to accept offloaded tasks from other devices, incentives may be given in the form of crypto-currency. The incentive scheme would reward participant nodes which deliver high performance to tasks offloaded to them.

Orsini et al. [45] also proposed offloading computing tasks to the mobile edge. In mobile cloud computing (MCC), devices are generally limited in resources such as energy, computing, storage causing them to offload certain resource-intensive tasks. However, latency-sensitive applications (e.g., cloud gaming, entertainment, virtual reality) require high-speed data communications which is not always possible with the distant cloud which led to the emergence of mobile edge computing (MEC). MEC actually falls under the umbrella of today's adoption of fog computing or edge computing. The authors interchangeably use the terms mist computing and cloudlets to refer to the aggregated pool of resources for mobile devices. Cisco's IOx network infrastructure products allow running third party virtual machine (VM)-based services directly on the infrastructure (such as routers) close to mobile devices. In this way, resource management can be performed at the edge of the end nodes. The authors do not elaborate how the resource management can be performed. However, they emphasize on a hierarchical offloading of tasks from mobile nodes to mist, from mist to fog, and eventually, from fog to cloud, all depending on the requirements of an application. The authors argue that we need a well-defined and rich hierarchy between the underlying end-nodes (called surrogates) and the service providers (such as cloud). In this work, the mist computing infrastructure, located between the fog and the end-nodes at the edge enables better decision making such as what to offload, rather than offloading unintelligently. Figure 4 further elaborates the concept presented by the authors.

Pu et al. [46] discussed task offloading for mobile users on the basis of incentive to the offloadee. The authors proposed a collaborative environment for mobile nodes, which they call as device to device (D2D) fogging. The proposed system relies on collaborative devices around the device that requires offloading of tasks in order to conserve energy. However, the control and decision about offloading is done by the network operators, such as through base station, which makes the system relatively complex and more interactive to the network causing more messages to be exchanged within the network. To avoid over-exploiting the offloadee, some incentive mechanism is incorporated by the mediating application, residing on both the devices

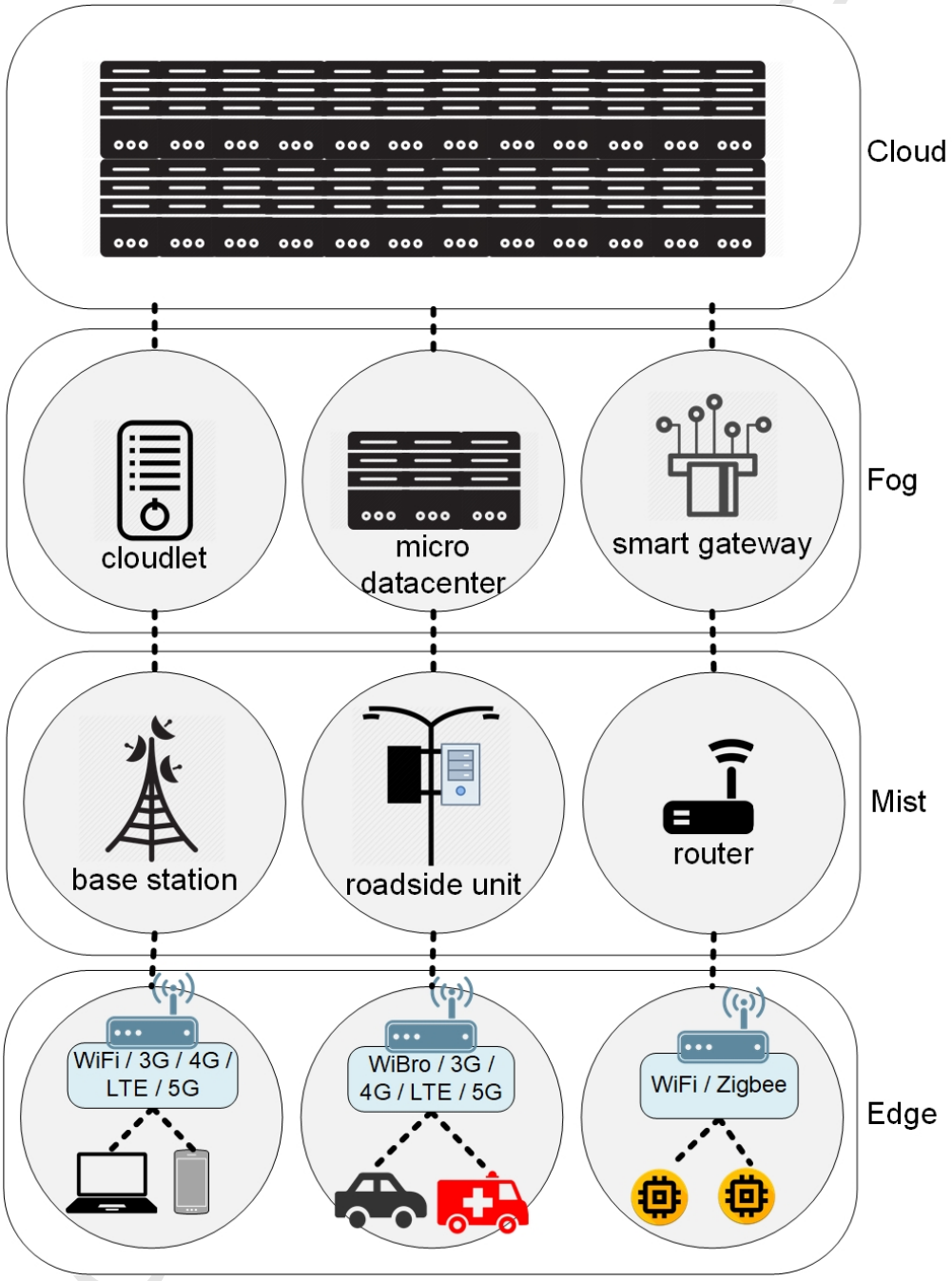

Figure 4: Mobile edge computing with different perceptions.

- the offloader and the offloadee. The incentive is based on tit-for-tat offloading mechanism. That is, when resources are contributed by a node the contributors account is credited. In the future, the device that sought resource offloading, will have to contribute as well for the device that ran its tasks previously. The contributions and usage of resources by a device are maintained by the network operator.

Meurisch et al. [47] identified a major challenge that relates to the deciding when to initiate task offloading decision. The authors state that normally tasks are offloaded to the surrogate nodes - the offloadees, that are already known and the network conditions such as delay and bandwidth between the offloading nodes. However, as nodes are mobile, they are often in the vicinity of unknown nodes. In this case, it is hard to make a context-aware decision. Hence, in their proposed methodology, the tasks are converted into micro-tasks and the unknown devices are then probed based on the set of micro-tasks. If the performance and network conditions are acceptable, medium or large sized tasks can be offloaded as well. Based on the analysis done on this approach, predictions of when to offload tasks based on micro-tasks have been more than 85% correct.

Zhang et al. [48] focused on the importance of energy consumption. They presented a methodology for energy-efficient computation offloading in 5G networks. Their proposed offloading system takes into account both the energy consumption of task execution, as well as the energy consumption incurred during the transmission or offloading of the task. The offloading system jointly optimizes the offloading of the task and radio resource allocation in 5G networks in order to minimize energy consumption of the overall offloading process. As wireless channel states vary and the offloaded tasks' sizes also vary, the energy consumption for transmission also changes accordingly. Moreover, since mobile devices share with each other the radio resources of the network, interference may also occur among each other. As a result, tasks transmission rates are decreased, and energy efficiency is negatively affected. In their approach, each task that is offloaded is assumed to be atomic and cannot be divided further into chunks. When a mobile node has to offload a task, it computes the energy consumption in two cases. The first case is if the task is executed locally, and the second case is if the task is executed at the base station. If the latter consumes less energy, the task is offloaded there. Otherwise, the task is executed locally.

Considering the importance of healthcare and the role of fog computing in it, Craciunescu et al. [49] proposed offloading of healthcare cloud data to

the edge to address latency issues. The experimentation, which is an e-health laboratory implementation, was conducted using a home PC and extracted data (all data, without any selection) from smartphones, MetaWear device, ProMove3D device, is sent to the cloud for further processing. They used fog computing for critical healthcare data, to provide a quick response to the patient. However, critical data is not further explained in their proposed approach. Data is gathered from various healthcare sensors, such as pulse rate, oxygen level. The data arrives at a middleware fog where algorithms are implemented to determine if a threshold (such as for instance, low oxygen level) regarding any sensor reading has been reached. The fog nodes trigger a notification to the user in the case of an emergency. According to the analysis presented, when the same task is executed by the cloud, the latency increases by 2 to 4 seconds compared with the fog computing middleware approach.

Liu et al. [50] also emphasized the importance of incentive-driven computation offloading because it would be difficult to encourage edge nodes to execute tasks on offloaders' behalf without any incentive. The interaction among cloud nodes and edge servers is formulated through a Stackelberg game aimed at maximizing cloud service utilities. Their analysis results in a unique Nash equilibrium. Their proposed approach addresses two main issues. First, when a cloud server receives a request to offload data to an edge server, the cloud server needs to determine whether to accept or reject the request. The second issue is how much to pay as an incentive to the surrogate edge node. To address this issue, they designed the Stackelberg game where an equilibrium is determined between the cloud and the edge nodes. In this case, offloading from the cloud is done based on the payment made against the offloaded tasks to the edge node.

Habak et al. [11] emphasized the need to harvest the unused computational capabilities of nearby mobile nodes thereby creating a femto cloud. Mobile nodes are becoming increasingly more powerful over time and in most cases, their resources are underutilized. Harvesting resources from nearby mobile nodes, such as in public transit, coffee shops, and other places where people gather can be very useful in terms of power efficiency, latency minimization, and computation efficiency. The edge application residing on the participating phones performs the offloading by seeking and granting permissions for task offloading. The incentive can be based on the location (e.g., coffee shop credits, transport credit, university credit, and so on) where people gather.

Wang et al. [51] proposed offloading tasks from mobile devices to the femto clouds. Femto access points (FAPs) are the entities that reside within the radio access network of mobile devices. FAPs collaborate to form the femto cloud. However, the authors proposed that, for strict latency constraints, applications may partially offload tasks, rather than performing a full offload to the femto cloud. In this way, from the benefits of parallelism (with parallel executing of tasks on femto cloud nodes) can be reaped. In this way, the communication bandwidth can be better managed because tasks are divided and then shared among the collaborating devices. To further enhance their system, the authors argued that simply offloading tasks does not yield overall efficiency. Rather, energy consumption should also be taken into account, for which they use dynamic voltage scaling (DVS). Through DVS, the computation speed of a mobile device can be made adaptive in order to reduce the energy consumption.

Table 2 provides a comprehensive picture of the reviewed papers.

Table 2: Analysis of offloading approaches proposed in the recent literature.

Technique	**Metrics**	**Results**	**Portability**	**Strengths**	**Weaknesses**
Zhao [41]	Energy and delay	Fog incurs up to 87% less energy consumption and energy cost, compared to cloud	High	Generic and easy to implement	Fog ownership and tasks are undefined
Continued on next page					

Table 2 – continued from previous page

Technique	Metrics	Results	Portability	Merits	Demerits
Liang [42]	SDN flow	Proposed architecture is implemented using SDN controller in a lab environment, but not evaluated	Low	Distinct flow for each application	Limited deployment, higher complexity
Fricker [43]	Load balancing	Up to 70% improvement in avoiding loss by timely offloading from an overloaded datacenter to another one	High	Minimize request rejection at overloaded datacenter	Deployable only in the case of service request congestion
Hasan [44]	Incentive, computation	66% less energy consumption, compared to without offloading scenario	Low	Can be easily implemented and deployed, incentive driven	Based on crypto currency, which is still maturing

Continued on next page

Table 2 – continued from previous page

Technique	Metrics	Results	Portability	Merits	Demerits
Orsini [45]	Computation	Present an architecture for hierarchical offloading, without evaluation	Medium	Hierarchical, good for complex applications	Limited use, high complexity
Pu [46]	Incentive, computation	Proposed offloading scheme save roughly 30%, 23%, and 18% energy over random, greedy, and reciprocal schemes, respectively	Low	Realistic, incentive driven	Additional processing and record maintenance at the provider's end are needed

Continued on next page

Table 2 – continued from previous page

Technique	Metrics	Results	Portability	Merits	Demerits
Meurisch [47]	Computation	Achieved an accuracy up to 85.5% after two micro task samples for predicting the runtime performance of unknown services	Medium	Assured to deliver high performance	Additional complexity in dividing tasks and monitoring performance
Zhang [48]	Energy, computation	18% decrease in energy consumption with the proposed strategy	Medium	Adaptively energy aware	Tasks are atomic and cannot be divided
Craciunescu [49]	Latency	Algorithm for the fall detection has a 90% accuracy	High	Tested on healthcare applications	Limited in scenario used in their evaluation

Continued on next page

Table 2 – continued from previous page

Technique	Metrics	Results	Portability	Merits	Demerits
Liu [50]	Incentive, computation	Offloading ratio increases with the incentive	High	Incentive driven	Payment undefined, complex to execute because highly dependent on the type of incentive
Habak [11]	Incentive, computation	85% improvement in performance	Low	Incentive driven, realistic	Evaluation and testing were done using a limited scenario
Wang [51]	Computation, energy consumption	36% less energy consumption compared to partial offloading	Low	Adaptive computation scaling based on performance	Complex to implement due to breakdown of tasks and partial offloading

7. Research Challenges

We discuss some of the challenges that need to be addressed in the future in order to enable seamless, efficient, reliable offloading of tasks in fog computing to deliver high end-to-end performance and low latency (as well as fast response) for a wide range of fog-based applications.

7.1. Resource allocation and scalability

One of the key challenges related to offloading is to find out the right amount of resources required at the location where the tasks will be executed.

It is well-known that if we have more resources, then some may be underutilized and if we have few resources, then offloading may be triggered too often. The tradeoff in the amount of resources will depend on the type of service. When the resources need to be increased to avoid offload, and when the resources should not be scaled up and offload should be encouraged remains a challenge.

7.2. Scalability of application

When an application is deployed, a certain number of users is anticipated. The ability to handle more users will depend on the capability of the background algorithms to handle more requests. The process that performs the offloading makes use of various parameters such as load balancing, energy efficiency, so on, as we discussed previously. Hence, how much the application should be scaled up so that the task of offloading is not affected, remains a challenge.

7.3. Service level agreement and compatibility of services

Offloading involves at least two entities: tasks to be offloaded and the provider of services where these tasks are offloaded, ensuring that the original service level agreement (SLA) is not violated during offloading remains a challenge. SLA matching, SLA monitoring and violation, are important factors that need to be considered in such case. Different devices would be running different algorithms or types of software, and will most likely execute tasks in different ways - which may not be compatible (e.g., meeting the latency requirement) with each other.

7.4. Security and privacy of data and users

When offloading occurs, the more data hops we have the great will be the risk of data theft and misuse. When multiple nodes and systems communicate with each other, data communication and storage are more prone to intrusion and theft. The data can be misused and may even result in huge losses, such as threat to life in a healthcare environment, manufacturing malfunctions in an industrial environment, accidents in an ITS, and so on. As a result, efficient and robust data security measures would be required, trustworthy entities have to be involved which means that trustworthiness parameters have to be defined and applied so that the decision on offloading is precise.

7.5. Integration and interoperablity

In a highly heterogeneous networked environment (with different types of hardware and software), interoperability arises. In the case of IoT-fog-cloud 3-tier communication, interoperability challenges increase. One example would be different storage technologies used by different cloud storage providers. Each storage service may have different compression mechanisms - affecting the size of the stored data, different synchronization mechanisms - affecting the duration of synchronization, and different data security and privacy mechanisms. A more in-depth discussion on heterogeneity issues is presented in [21] in which different cloud storage providers (such as Dropbox, Google Drive, Box, Microsoft OneDrive, SugarSync, and Amazon CloudDrive) are analyzed on the aforementioned parameters.

7.6. Fault tolerance and reconfiguration of offloading system

When additional entities, other than the one hosting the service, are involved in the offloading process, when a fault arises at the secondary offloaded device (e.g., cloud, when data is offloaded by a fog) then how fog is going to reconfigure and execute alternate steps remains an open issue. As the offloading system becomes increasingly automated and heterogeneous entities are involved, the risks of failure increase. Device malfunction, delayed communication, and connectivity failures are some common examples. The offloading process must be robust and be capable of not only detecting and withstanding common faults but also be capable of detecting and handling faults in a timely manner. Advanced fault detection algorithms will have to be applied at the hub, gateway, or middleware that is responsible for coordinating different machines and devices. In such cases, the accuracy and timeliness of these algorithms in detecting fault become increasingly important.

7.7. Energy consumption tradeoff

Offloading tasks on its own is an energy and bandwidth consuming process. This means, although largely inevitable, it would sometimes remain a tradeoff whether to go for offloading or not. One such example is the Camera Upload feature of Dropbox, which by default, uploads every photo taken to the Dropbox thereby resulting in an increase in battery consumption as well as network bandwidth.

7.8. Incentives for offloading service

As offloading becomes more of a service type offered in fog computing, it may become a liability in some sense and one should therefore compensate the entity performing the offloading tasks with some reasonable incentives, such as through crypto-currency payments [44] or air-miles-like rewards.

7.9. Monitoring

Offloading to different entities and especially, when incentives are involved, means that it would be necessary to monitor the service quality, delivery, robustness, and fault analysis. Hence, monitoring of the overall offloading process would require accurate and well-proven methods.

8. Conclusion

Proliferation of IoT devices, mobile computing, and communication technology coupled with intelligent services have contributed to the emergence of more interactive and autonomous services. Computation is becoming cheaper and more accessible than before although often largely available in chunks. This means that to execute certain tasks, offloading can be performed as a new service paradigm in cloud-IoT ecosystem. Task offloading can take place for various reasons, such as load balancing, latency mitigation, security, data management, energy management, and so on. Offloaded tasks directly impact the scalability of the physical resources as well as the service itself. Hence, we often need to decide on the tradeoff between when and how much to offload. In this paper, we discussed different types of offloading techniques that have been recently proposed in the literature focusing on the fog or edge computing in the cloud-IoT environment. We also presented some of the criteria that have been proposed when determined when to offload either to the cloud or to the fog. We reviewed some of the technologies (such as wireless communication, virtualization, AI) that are being deployed to enable offloading. We have also presented different offloading scenarios described in the literature and we discussed recently proposed offloading approaches in for such scenarios. Finally, we highlighted some key research challenges associated with tasks offloading in fog computing.

Acknowledgement

This publication was made possible by NPRP grant # 8-1645-1-289 from the Qatar National Research Fund, a member of Qatar Foundation. The statements made herein are solely the responsibility of the authors.

We thank the anonymous reviewers for their valuable comments and suggestions which helped us to improve the content, organization, quality, and presentation of this paper.

References

[1] A. Saeed, A. Abdelkader, M. Khan, A. Neishaboori, K. A. Harras, A. Mohamed, Argus: realistic target coverage by drones., in: IPSN, 2017, pp. 155–166.

[2] A. Essameldin, K. A. Harras, The hive: An edge-based middleware solution for resource sharing in the internet of things, in: Proceedings of the 3rd Workshop on Experiences in the Design and Implementation of Smart Objects, ACM, 2017.

[3] M. Ibrahim, M. Gruteser, K. A. Harras, M. Youssef, Over-the-air tv detection using mobile devices, in: ICCCN, IEEE, 2017, pp. 1–9.

[4] A. B. Said, A. Mohamed, T. Elfouly, K. Harras, Z. J. Wang, Multimodal deep learning approach for joint eeg-emg data compression and classification, in: Wireless Communications and Networking Conference (WCNC), 2017 IEEE, IEEE, 2017, pp. 1–6.

[5] A. Emam, A. Mtibaa, K. A. Harras, A. Mohamed, Adaptive forwarding of mhealth data in challenged networks, in: e-Health Networking, Applications and Services (Healthcom), 2017 IEEE 19th International Conference on, IEEE, 2017, pp. 1–7.

[6] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, L. Sun, Fog computing: Focusing on mobile users at the edge, arXiv preprint arXiv:1502.01815.

[7] H. Abdelnasser, K. A. Harras, M. Youssef, Ubibreathe: A ubiquitous non-invasive wifi-based breathing estimator, in: MobiHoc, ACM, 2015, pp. 277–286.

[8] H. Abdelnasser, M. Youssef, K. A. Harras, Wigest: A ubiquitous wifi-based gesture recognition system, in: Computer Communications (INFOCOM), 2015 IEEE Conference on, IEEE, 2015, pp. 1472–1480.

[9] H. Abdelnasser, M. Youssef, K. A. Harras, Magboard: Magnetic-based ubiquitous homomorphic off-the-shelf keyboard, in: SECON, IEEE, 2016, pp. 1–9.

[10] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for vm-based cloudlets in mobile computing, IEEE pervasive Computing 8 (4).

[11] K. Habak, M. Ammar, K. A. Harras, E. Zegura, Femto clouds: Leveraging mobile devices to provide cloud service at the edge, in: Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on, IEEE, 2015, pp. 9–16.

[12] H. Gedawy, S. Tariq, A. Mtibaa, K. A. Harras, Cumulus: A distributed and flexible computing testbed for edge cloud computational offloading, in: Cloudification of the Internet of Things (CIoT), IEEE, 2016, pp. 1–6.

[13] A. Mtibaa, A. Emam, S. Tariq, A. Essameldin, K. A. Harras, On practical device-to-device wireless communication: A measurement driven study, in: Wireless Communications and Mobile Computing Conference (IWCMC), 2017 13th International, IEEE, 2017, pp. 409–414.

[14] A. Mtibaa, K. A. Harras, K. Habak, M. Ammar, E. W. Zegura, Towards mobile opportunistic computing, in: Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on, IEEE, 2015, pp. 1111–1114.

[15] A. Saeed, M. Ammar, K. A. Harras, E. Zegura, Vision: The case for symbiosis in the internet of things, in: Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services, ACM, 2015, pp. 23–27.

[16] Y. Jararweh, A. Doulat, O. AlQudah, E. Ahmed, M. Al-Ayyoub, E. Benkhelifa, The future of mobile cloud computing: integrating cloudlets and mobile edge computing, in: Telecommunications (ICT), 2016 23rd International Conference on, IEEE, 2016, pp. 1–5.

[17] A. Elgazar, K. A. Harras, M. Aazam, Towards intelligent edge storage management: Determining and predicting mobile file popularity, in: Mobile Cloud (Mobile CLOUD), 2018 IEEE 6th International Conference on, IEEE, 2018.

[18] X. Masip-Bruin, E. Marín-Tordera, G. Tashakor, A. Jukan, G.-J. Ren, Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems, IEEE Wireless Communications 23 (5) (2016) 120–128.

[19] C. Huang, R. Lu, K.-K. R. Choo, Vehicular fog computing: architecture, use case, and security and forensic challenges, IEEE Communications Magazine 55 (11) (2017) 105–111.

[20] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K.-K. R. Choo, M. Dlodlo, From cloud to fog computing: A review and a conceptual live vm migration framework, IEEE Access 5 (2017) 8284–8300.

[21] M. Aazam, E.-N. Huh, M. St-Hilaire, Towards media inter-cloud standardization–evaluating impact of cloud storage heterogeneity, Journal of Grid Computing (2016) 1–19.

[22] S. Ibrahim, H. Jin, B. Cheng, H. Cao, S. Wu, L. Qi, Cloudlet: towards mapreduce implementation on virtual machines, in: Proceedings of the 18th ACM international symposium on High performance distributed computing, ACM, 2009, pp. 65–66.

[23] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, et al., Mobile-edge computing introductory technical white paper, White Paper, Mobile-edge Computing (MEC) industry initiative.

[24] K. Al Agha, G. Pujolle, T. Ali-Yahiya, Mobile-edge computing, Mobile and Wireless Networks 283–306.

[25] A. Alsaffar, M. Aazam, C. S. Hong, E.-N. Huh, An architecture of iptv service based on pvr-micro data center and pmipv6 in cloud computing, Multimedia Tools and Applications 76 (20) (2017) 21579–21612.

[26] M. Aazam, E.-N. Huh, Dynamic resource provisioning through fog micro datacenter, in: Pervasive Computing and Communication Workshops

(PerCom Workshops), 2015 IEEE International Conference on, IEEE, 2015, pp. 105–110.

[27] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, P. Rodriguez, Greening the internet with nano data centers, in: Proceedings of the 5th international conference on Emerging networking experiments and technologies, ACM, 2009, pp. 37–48.

[28] K. Fall, S. Farrell, Dtn: an architectural retrospective, IEEE Journal on Selected areas in communications 26 (5).

[29] S. Jain, K. Fall, R. Patra, Routing in a delay tolerant network, Vol. 34, ACM, 2004.

[30] C. Sobin, V. Raychoudhury, G. Marfia, A. Singla, A survey of routing and data dissemination in delay tolerant networks, Journal of Network and Computer Applications 67 (2016) 128–146.

[31] S. Kabadayi, A. Pridgen, C. Julien, Virtual sensors: Abstracting data from physical sensors, in: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks, IEEE Computer Society, 2006, pp. 587–592.

[32] S. Madria, V. Kumar, R. Dalvi, Sensor cloud: A cloud of virtual sensors, IEEE software 31 (2) (2014) 70–77.

[33] J. Zhou, D. Gao, D. Zhang, Moving vehicle detection for automatic traffic monitoring, IEEE transactions on vehicular technology 56 (1) (2007) 51–59.

[34] D. Zhao, Y. Dai, Z. Zhang, Computational intelligence in urban traffic signal control: A survey, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42 (4) (2012) 485–494.

[35] M. Aazam, E.-N. Huh, E-hamc: Leveraging fog computing for emergency alert service, in: Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on, IEEE, 2015, pp. 518–523.

[36] K. Kramer, D. Hedin, D. Rolkosky, Smartphone based face recognition tool for the blind, in: Engineering in Medicine and Biology Society

(EMBC), 2010 Annual International Conference of the IEEE, IEEE, 2010, pp. 4538–4541.

[37] S. Zeadally, R. Hunt, Y.-S. Chen, A. Irwin, A. Hassan, Vehicular ad hoc networks (vanets): status, results, and challenges, Telecommunication Systems 50 (4) (2012) 217–241.

[38] A. P. Jayasumana, Q. Han, T. H. Illangasekare, Virtual sensor networks-a resource efficient approach for concurrent applications, in: Information Technology, 2007. ITNG'07. Fourth International Conference on, IEEE, 2007, pp. 111–115.

[39] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, P. Polakos, Wireless sensor network virtualization: A survey, IEEE Communications Surveys & Tutorials 18 (1) (2016) 553–576.

[40] J. Contreras-Castillo, S. Zeadally, J. Guerrero, Internet of vehicles: Architecture, protocols, and security, IEEE Internet of Things.

[41] X. Zhao, L. Zhao, K. Liang, An energy consumption oriented offloading algorithm for fog computing, in: International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, Springer, 2016, pp. 293–301.

[42] K. Liang, L. Zhao, X. Chu, H.-H. Chen, An integrated architecture for software defined and virtualized radio access networks with fog computing, IEEE Network 31 (1) (2017) 80–87.

[43] C. Fricker, F. Guillemin, P. Robert, G. Thompson, Analysis of an offloading scheme for data centers in the framework of fog computing, ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS) 1 (4) (2016) 16.

[44] R. Hasan, M. Hossain, R. Khan, Aura: An incentive-driven ad-hoc iot cloud framework for proximal mobile computation offloading, Future Generation Computer Systems.

[45] G. Orsini, D. Bade, W. Lamersdorf, Computing at the mobile edge: Designing elastic android applications for computation offloading, in: IFIP Wireless and Mobile Networking Conference (WMNC), 2015 8th, IEEE, 2015, pp. 112–119.

[46] L. Pu, X. Chen, J. Xu, X. Fu, D2d fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted d2d collaboration, IEEE Journal on Selected Areas in Communications 34 (12) (2016) 3887–3901.

[47] C. Meurisch, J. Gedeon, T. A. B. Nguyen, F. Kaup, M. Muhlhauser, Decision support for computational offloading by probing unknown services, in: Computer Communication and Networks (ICCCN), 2017 26th International Conference on, IEEE, 2017, pp. 1–9.

[48] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, Y. Zhang, Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks, IEEE Access 4 (2016) 5896–5907.

[49] R. Craciunescu, A. Mihovska, M. Mihaylov, S. Kyriazakos, R. Prasad, S. Halunga, Implementation of fog computing for reliable e-health applications, in: Signals, Systems and Computers, 2015 49th Asilomar Conference on, IEEE, 2015, pp. 459–463.

[50] Y. Liu, C. Xu, Y. Zhan, Z. Liu, J. Guan, H. Zhang, Incentive mechanism for computation offloading using edge computing: A stackelberg game approach, Computer Networks 129 (2017) 399–409.

[51] Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li, Mobile-edge computing: Partial computation offloading using dynamic voltage scaling, IEEE Transactions on Communications 64 (10) (2016) 4268–4282.

Mohammad Aazam S'11, M'15, SM'16 is currently working as a postdoc research associate at Carnegie Mellon University - Qatar since July 2017 and a postdoctoral fellow at Ryerson University, Canada since March 2017. Previously, he has been a sessional faculty and postdoc fellow with Carleton University, Canada from July 2015-2017. He is also working as a postdoc collaborator with University of Kentucky, USA and University of Ontario Institute of Technology (UOIT), Canada. He completed his Ph.D. Com-puter Engineering from Kyung Hee University, Korea, in 2015. In addition to that, he has completed a course on Data Science with R from Harvard University, USA in 2017, in which he topped the course securing 100% score. He also completed a course on Internet of Things (IoT) from King's College London, UK, in 2016. He started his professional career in 2006, serving as a university lecturer and a researcher, working on several R&D projects. He is currently serving IEEE Comnunications Magazine, Sensors & Transducers journal, and Elsevier Digital Communications & Networks journal as an editor/associate editor. He has served several conferences and workshops as a general chair, invited speaker, and invited session chair (IEEE Globecom '14, IEEE ICC '15, IEEE ICWS '15, IEEE MASS '16, IEEE WMNC '16, and IEEE CCNC '16). He has more than 90 publications, including 3 patents. He is also a recipient of IEEE AIYEHUM 2016 award. Aazam is a senior member of the IEEE. For additional details: www.aazamcs.com.

The digital world is expanding rapidly and advances in networking technologies such as 4G long-term evolution (LTE), wireless broadband (WiBro), low-power wide area networks (LPWAN), 5G, LiFi, and so on, all of which are paving the way for the emergence of sophisticated services. The number of online applications is increasing along with more computation, communication, and intelligent capabilities. Although current devices in use today are also getting more powerful in terms of features and capabilities, but they are still incapable of executing smart, autonomous, and intelligent tasks such as those often required for smart healthcare, ambient assisted living (AAL), virtual reality, augmented reality, intelligent vehicular communication, and in many services related to smart cities, Internet of Things (IoT), Tactile Internet, Internet of Vehicles (IoV), and so on. For many of these applications, we need another entity to execute tasks on behalf of the user's device and return the results - a technique often called offloading, where tasks are outsourced and the involved entities work in tandem to achieve the ultimate goal of the application. Task offloading is attractive for emerging IoT and cloud computing applications. It can occur between IoT nodes, sensors, and devices and the edge nodes or fog nodes. Offloading can be performed based on different factors that include computational requirements of an application, load balancing, energy management, latency management, and so on. We present a taxonomy of recent offloading schemes that have been proposed for domains such as fog, cloud computing, and IoT. We also discuss the middleware technologies that enable offloading in a cloud-IoT case and the factors that are important for offloading in a particular scenario. We also present research opportunities concerning offloading in fog and edge computing.