

Demystifying Fog Computing: Characterizing Architectures, Applications and Abstractions

Prateeksha Varshney and Yogesh Simmhan

Department of Computational and Data Sciences

Indian Institute of Science (IISc), Bangalore 560012, India

Email: prateeksha@grads.cds.iisc.ac.in, simmhan@cds.iisc.ac.in

Abstract

Internet of Things (IoT) has accelerated the deployment of millions of sensors at the edge of the network, through Smart City infrastructure and lifestyle devices. Cloud computing platforms are often tasked with handling these large volumes and fast streams of data from the edge. Recently, Fog computing has emerged as a concept for low-latency and resource-rich processing of these observation streams, to complement Edge and Cloud computing. In this paper, we review various dimensions of system architecture, application characteristics and platform abstractions that are manifest in this Edge, Fog and Cloud eco-system. We highlight novel capabilities of the Edge and Fog layers, such as physical and application mobility, privacy sensitivity, and a nascent runtime environment. IoT application case studies based on first-hand experiences across diverse domains drive this categorization. We also highlight the gap between the potential and the reality of Fog computing, and identify challenges that need to be overcome for the solution to be sustainable. Together, our article can help platform and application developers bridge the gap that remains in making Fog computing viable.

1

1 Introduction

The prime drivers of Big Data over the past decade have been the WWW, social media, eCommerce, and enterprise information systems, with data and services being consolidated in public and private data centers. However, *sensing at the edge* of the network is accelerating many-fold. This is enabled through infrastructure like Smart Cities that are being deployed globally, lifestyle devices like wearables and smart appliances, and observations collected *ad hoc* through crowd-sourcing over smart phones [3, 23]. A large class of this data streaming from the edge can be attributed to the *Internet of Things (IoT)*, where the

¹Pre-print: To appear in ICFEC-2017 proceedings

convergence of commodity sensors, accessible communications, and the need for intelligent infrastructure management is driving this exponential growth.

The traditional model of acquiring and processing this data from the edge to offer useful services has been through *Cloud computing*. These large Cloud data centers host computing and storage infrastructure with high bandwidth connectivity, and allow IoT applications and services to be hosted remotely [30]. Their on-demand access to seemingly infinite resources accessed through simple web service APIs, combined with the pay-as-you-go billing model that capitalizes on economies of scale have made Cloud computing popular for supporting millions of clients at the edge.

At the same time, two key challenges emerge with this hub-and-spoke model. Public Cloud data centers are globally distributed, but typically limited to one or a few per country (depending on the size). This means that the network distance from the edge to the Cloud results in round trip time (RTT) are in the order of 100's of milliseconds, just in terms of latency [20]. Many IoT applications are *latency sensitive*, be they demand prediction for Smart Power Grids [2,12] or voice responses in Apple Siri. Consequently, hosting the analytics and decision making in the Cloud can compromise their performance [6, 8].

Similarly, video is emerging as a major source of data from the edge, from urban surveillance and ATM cameras that are part of the infrastructure, to personal body cameras and crowd-sourced recordings from phones [23, 31]. Here, the *bandwidth* required to push high fidelity video streams to the Cloud is prohibitive, and this further increases the RTT if the video analytics on the Cloud has to control edge devices, like zooming a PTZ Camera or detecting faces in real-time.

This has led to a design paradigm of processing at the edge to supplement, or replace, Cloud computing [5, 10]. Also called *Edge Computing* or *Mobile Cloud* [30,31], these are typically used in vertically integrated applications where a part of the processing and analytics happens on the edge device while the Cloud is used for coordination and data archival. For e.g., a FitBit fitness watch may pre-process and visualize data on the smart phone before archiving to the Cloud.

However, Edge computing suffers from several deficiencies. The edge platforms tend to be constrained devices, with battery capacity or memory often being the limiting factor rather than even compute capability. This can cause resource contention if multiple IoT applications need to be deployed concurrently [8]. Also, there are no well-defined or robust runtime or management platforms for composing generic Edge computing applications, comparable to the virtualization or service-based architectures exposed in Cloud computing. As a result, edge computing is limited to bespoke solutions.

Fog computing [6], also known as *Cloudlets* [20], were introduced by Satyanarayanan, et al., and popularized by Cisco as a complementary resource-rich layer that sits between the edge device and the Cloud. [20] suggest that this layer be one network hop away from the edge to offer both low and predictable latencies to support gaming and video conference services. The Cloudlets are meant to serve as small-scale data centers that are placed closer to the edge [5],

where often both the generators of the observations and the consumers of the analytics reside. At the same time, the Fog layer can offer easier manageability of resources as services, and possibly a feasible business model similar to the popular Clouds.

In this regard, Fog computing has similarities with *Content Distribution Network (CDN)*, that sit close to the edge for low latency delivery of content, typically static or slow changing and populated from the Cloud, but are otherwise passive hosts. [23] considers Cloudlets as a “CDN in reverse”, with the edge populating the Cloudlet, and the Cloudlet serving data to the Cloud. At the same time, the Fog can also actively host services and analytics – closer to the edge than, even, CDN.

From a societal perspective, Fog computing also becomes relevant when there are network outages that restrict or remove the ability to reach the Cloud data center. For e.g., the *Cyclone Vardah* that hit Chennai, India in Dec. 2016 damaged a trans-oceanic optical cable connecting India with Singapore and Europe, where many commercial data centers are located ². In future, as critical services such as water, power and transport are controlled by smart algorithms, hosting them in a Fog layer with peering mechanism allows a graceful degradation of functionality even in the absence of access to the Cloud, rather than cause a debilitating loss of city services.

However, it is important neither to over-state the potential of Fog computing nor to dismiss it as hype. Despite the existence of the concept for several years now, commercial deployments of Fog Computing are yet to take off. Part of the reason is that there are still an inadequate number of widely deployed or critical applications that find the Fog to be essential. There is also lack of clarity on the application model, runtime and management environments for a Fog platform. And lastly, a sustainable business model and service providers are still evolving. So it is useful to take a *critical look* at Fog computing as a *prospective* pervasive platform.

In this paper, we attempt to characterize some of the key features of the system design and computing architecture of Fog computing; the application features, particularly from IoT use-cases, that motivate the need for Fog computing; and lastly the programming models and abstractions that can be leveraged to bridge the gap between the applications and the systems for designing intuitive runtime platforms.

Several papers offer desiderata for Fog computing and its concepts. Many tend to be superficial or narrowly defined. Cisco subsumes the Fog into the Edge (or vice versa), there by discounting widely-available devices like smart phones and Raspberry Pis deployed as part of Smart City infrastructure [6]. [28] attempts to define Fog computing, but views it from the limited prism of network management and connectivity. Several papers on Cloudlets offer exemplar applications that are actually deployed, but fail to offer an overarching technical view of this space [20, 21, 23]. Others propose a programming model

²A huge storm has messed up India’s Internet by Rishi Iyengar, CNN, December 14, 2016 <http://money.cnn.com/2016/12/14/technology/india-cyclone-varDAH-chennai-internet/>

for composing applications that run across mobile, Fog and Cloud layers as a Platform as a Service (PaaS), but limit it to a rigid hierarchy [11]. [15] attempts a similar effort as our paper, but favor mobile devices rather than edge devices at large. The latter, many of which can be part of an IoT infrastructure, are key. [8] discuss the role of Fog computing on IoT applications, and highlight open issues. These are further discussed in the related work section (§ 5).

Here, however, we go a step further and provide intrinsic dimensions and capabilities of Fog computing that should be of interest to platform developers and application designers, while recognizing that this is still an emerging space. These features distinguish the Fog from traditional mobile or Cloud computing but *form a continuum*. These characteristics are as yet inadequately studied in the context of a holistic Edge, Fog and Cloud computing eco-system. We also motivate the problem space with emerging, rather than futuristic, applications.

We make the following specific contributions in this paper:

1. We characterize the *expected capabilities of a Fog computing system*, and lay them in the context of existing Edge/Mobile and Cloud computing architectures. We highlight dimensions that distinguish these alternative, but inter-linked, resource layers (§ 2).
2. We review the *application requirements* that motivate the need for a Fog layer, and identify the gaps posed by a Cloud-only or Edge+Cloud model. We also offer case studies of applications and their features (§ 3).
3. We highlight the possible *runtime and middleware capabilities, and application models* required in a Fog layer, while recognizing that the Fog platform will need to adapt to the unique needs of applications and the actual manifestation of the Fog system (§ 4).
4. Lastly, we *discuss* the gap between potential and reality of Fog computing, and identify challenges that need to be overcome and solutions that are possible in this evolutionary phase (§ 6).

2 Fog Computing System Architecture

2.1 Definitions

There exist various overlapping definitions of what Fog computing is, in literature. We first summarize some of these views, and then take up a more detailed characterization.

- Satyanarayanan, et al, [19] state that Cloudlets, synonymous with Fog computing [22], are a resource-rich (cluster of) computers that are located one hop away from the mobile devices. They have Gigabit interconnect, and high bandwidth, through Wireless to the mobile device and through WAN to the Internet. The Cloudlets offer a “data center in a box” close to the mobile device in order to reduce the network latency and bandwidth-induced latency to support interactive applications.

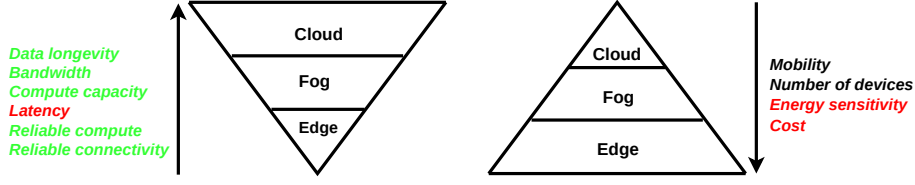


Figure 1: Resource characteristics of Cloud, Fog and Edge computing systems

- Cisco [6] views Fog Computing as offering resources like compute, storage and networking similar to Clouds, with support for virtualization and multi-tenancy. These are geo-distributed to offer low and predictable latencies to client applications that are mobile or part of the infrastructure. They do conceive of the Fog as having modest resource capabilities, ranging from an edge network router to a high-end server.
- [28] define Fog computing as a large collection of heterogeneous and decentralized devices, communicating among themselves to store data and process tasks, that can be leased to users to support basic network functions or sandboxed applications. They take a communication-centric view of Fog computing, with 4G/5G connectivity enabling the Fog layer and Software Defined Network (SDN) management pushed to the Fog.

2.2 Comparing Edge, Fog and Cloud Computing

From the above definitions, among many others, Fog computing is seen as a resource layer that fits between the edge devices and the Cloud data centers, with features that may resemble either. Hence, it is worth comparing and contrasting the characteristics of the Fog computing system architecture relative to the capabilities of Edge and Cloud computing.

2.2.1 Resource Characteristics

There are several resource and performance characteristics that distinguish these three layers, as captured in a pyramid structure in Fig. 1. Chief among them, serving as the fundamental motivation, are the *network characteristics*. The Fog is close to the edge in the network topology. Hence, it has a lower latency for access in *both* directions, i.e., serving content from Fog to Edge, and pushing data from Edge to Fog [19]. Whether it is at a 1-hop or multi-hop depends on the deployment. Further, the bandwidth between the Edge and Fog is also higher (e.g., using WiFi) than from the Edge to the Cloud (e.g., using cell networks). The Fog is also expected to have a high-bandwidth and reliable link to the Internet [26]. The connectivity between the Edge and the Fog may be less robust than between the Fog and the Cloud due to the use of wireless links for the last mile [15].

In these regards, the Fog layer is analogous to a CDN but typically closer to the Edge [28]. The *storage capacity and data longevity* of Fog layer is much

higher than the Edge devices, though more limited than Clouds. This storage can be used to cache large datasets that are useful to edge devices, such as apps or Virtual Machine (VM) images that have to be installed on devices for Smart City infrastructure, or snapshots of historic data [5]. But more interestingly, the Fog serves as a “reverse CDN” to allow edge devices to push data to the Cloud [7]. This allows scenarios where data is staged in the Fog and periodically pushed to the Cloud for archival, potentially after some pre-processing such as deduplication [9, 23]. Further, edge devices can also access data pushed up to the Fog by other edge devices as well [15].

At the same time, the Fog layer also offers *compute resources* that have a higher capacity and reliability than the Edge but to a smaller scale than Cloud data centers [6, 27]. This resource capability can be used to host applications and services that range from Video Analytics as a service for processing video frames close to the Edge or directory services where devices register and access location-sensitive configuration for mobile phones [14].

The sheer number of Fog appliances will dwarf the number of Cloud data centers (even if not their cumulative compute power), just as the number of edge devices number in the billions. This makes *manageability* of the Fog more challenging, but easier than the Edge. The economies of scales will also come into play if Fog hardware is standardized, similar to commodity smart phones or shipping containers with Cloud hardware. This can make Fog computing affordable and (if the edge device is not captive) cheaper than Edge computing, though Clouds will retain their *pricing* advantage.

Lastly, the *energy profile* can influence the capability and availability of some resources. Edge devices are often concerned with battery life, and the choice of using specific Edge features may depend on current battery level [17]. Cloud data centers reduce their energy footprint, but to limit operational costs [5]. The Fog layer is expected to run off grid power and, like the Cloud, be conscious of energy use to ensure fair pricing [9, 14, 28]. But there may be cases of remote deployments where the Fog may run off renewables like solar where energy conservation may be a primary goal.

2.2.2 Physical Presence and Access

Besides network distance, it is also worth considering the physical distance between the three computing paradigms, and their accessibility by clients. In Fig. 2, four quadrants are formed from considering whether the resources within a layer are *physically centralized or distributed* (*Y Axis*), and whether their *access is global or restricted* (*X Axis*).

Resources in a Cloud data center are centrally located, but depending on whether the Cloud is public or private, are available to any client in a pay-as-you-go model, or only to users who are part of the private corporation [15]. That said, public Cloud providers host multiple data centers that are geographically distributed, sometimes several in a country or continent, while the number of large private data centers for an enterprise tends to be more limited.

We can compare Edge and Fog resource distribution and access along a simi-

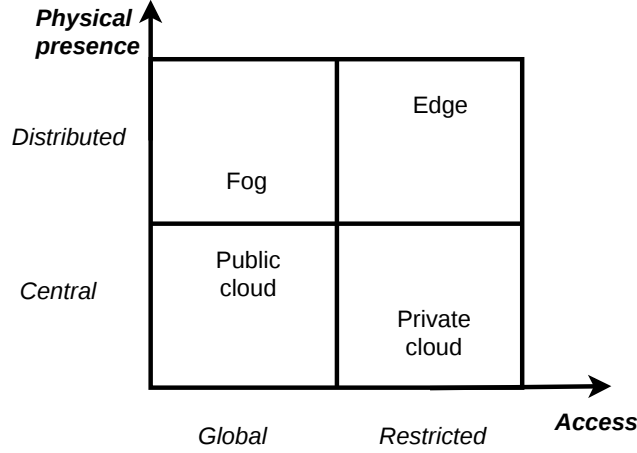


Figure 2: Physical presence vs access of Cloud, Fog and Edge resources

lar scale. Edge resources such as smart phones and set top boxes are distributed far and wide, but the access to them are restricted to individual users or managed applications [6,31]. Fog resources are also geographically distributed to be close to the edge, but not as dispersed though much more than Cloud data centers. Additional specializations, on whether there is a Fog for each city block, one for the whole city or other variants, depend on the business models and applications that will evolve. One also expects the Fog to offer as a shared, pay-as-you-go IaaS or PaaS model [5,31].

The distributed nature of Edge and Fog resources increases their *attack and failure surface*. There is a high chance of some device or Fog server failing or a network link dropping, and resiliency has to be built into the platform and application if necessary [16]. Clouds being more centralized are single points of failures, but Cloud fabrics and Big Data platforms internalize faults within the data center. But, as mentioned, natural disasters can cause these central hubs to get islanded, and bugs and security breaches can cause massive data loss.

The access restrictions on private Clouds and Edge devices translates to a *zone of trust* for applications and services hosted on them, which enables sensitive data and services to be hosted on them. Fog and public Clouds, however, are designed as shared resources with *multi-tenancy*, which require higher measures of security and sand-boxing between different applications or users, using containers or hypervisors [8,28].

That said, there may be Fog architectures where the resources are deployed for specific applications or organization (e.g., a Smart City municipality) and not available for rent, similar to a private Cloud [15]. Further, the Fog layer may sit at the boundary between the public and private network. For e.g., as part of the IISc Smart Campus project ³, there are hundreds of edge devices and sensors connected to the campus LAN that monitor the water, power and road

³IISc Smart Campus Project, <http://smartx.cds.iisc.ac.in>

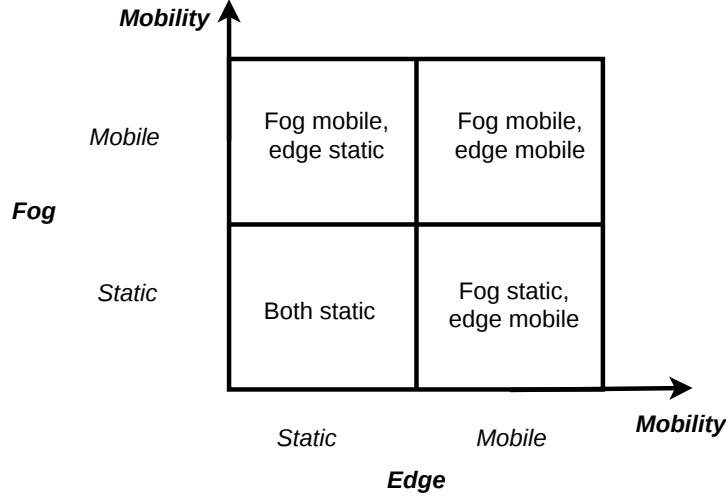


Figure 3: Physical mobility of Fog and Edge resources

network, and Fog servers on the DMZ are connected both to the private LAN and to the public WAN. This makes them well-suited to run proxy services that translate from one zone of trust to another, one service layer to another (e.g., CoAP to HTTP), or one network protocol to another (e.g., IPv6 to IPv4).

2.2.3 Mobility

It helps to understand the impact of mobility on these three resource layers as this impacts the communications, applications and platform design. We distinguish between mobility of the *physical resource*, discussed here, and mobility of the logical applications, which we examine later.

Cloud data centers, obviously, are not mobile though their platforms can ease the mobility of data and applications among their geo-distributed data centers. However, spatial mobility at the edge layer is frequent though not universal. Edge mobility is exemplified in ubiquitous smart phones, connected vehicles, and drones, while they remain static in, say, traffic cameras and smart power meters [14, 27]. Likewise, the Fog layer can also be manifest as a static or mobile resource platform [15]. A Fog server can be installed at fixed sites such as a coffee shop or the airport, or on mobile vehicles such as taxi cabs or trains. Fig. 3 captures these scenarios.

This mobility can have consequences on the link between the Edge and the Fog, or from the Edge to the public or private network, and these depend on the communication protocols used. However, we need to recognize that mobility can cause the network connectivity to be intermittent [24]. This can cause a transient loss of access to data or compute resources between the Edge and the Fog or the Cloud, and the Fog and the Cloud. But, in technologies where the access is based on physical proximity, such as Bluetooth, Near Field

Communication (NFC) or emerging line-of-sight technologies like Millimeter Wave, the disconnection can be permanent [28].

While we do not expect this ephemeral behavior for links between the Fog and the Cloud, such cases are possible in interaction among edge devices or with the Fog. For e.g., we use the notion of a “Data Sherpa” [17] to transfer observed data from a static sensor (edge) to a mobile smart phone (edge) using Bluetooth when they are close by, and then use the smart phone to push the data to the Fog or Cloud. But the interaction between the two edges is opportunistic, not planned or repeatable [24]. Similarly, users in a train can use their personal devices with the Fog servers in the coach to access media or make use of a high-speed Internet, but only while they are riding. Connected cars (Fog) can peer with nearby vehicles to share information on road condition and collaboratively navigate traffic [15]. As a result, depending on the mobility of these two layers, the application and platform will need to be designed based on *permanent, transient, periodic, or ephemeral connectivities* between the layers and within the layers which can determine the *reliability of access* to data, storage, network and computing resources.

Lastly, applications may also relate spatial proximity with *trust and context*. Fog layers physically close to the edge may be trusted by the edge applications to share sensitive data or for the Fog to offer services. This is already used to start cars using a “key fob” present in the car rather than a key in the ignition, or using NFC to share contacts or Bluetooth to pair devices. This can be leveraged more actively by applications for trusted interactions between the Edge and Fog layers. This closeness also offers context for the interaction and can be used to serve content relevant to the person from the Fog [31].

3 Applications for Fog Computing

In this section, we go in-depth into the characteristics and requirements of applications that are well-suited to different forms of Fog resources, in concert with the Edge and Cloud resources. We also offer detailed use cases on Fog applications.

3.1 Mobility of Data and Analytics

Similar to physical mobility of the resources, we can have logical movement of applications between Edge, Fog and Cloud layers. The entity that moves can be a *process* (e.g., analytics, services), *data* (e.g., video streams, device images) or *event* (e.g., control signals, emergency notifications). Each have their own requirements from the resources.

Processes require appropriate execution environments in the resource layer and adequate compute and memory capacity. While virtualized Clouds offer the most flexibility in this regard with various VM images and sizes available, Fog is expected to have a similar flexibility using containers or hypervisors [9,30]. Edge may be the most constrained in meeting dependencies, and the choices may be

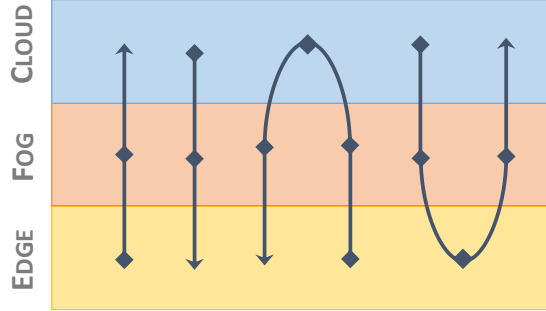


Figure 4: Logical mobility of applications between Edge, Fog and Cloud layers. Arrows indicate direction, diamonds indicate optional processing or staging at that layer. Process, Data and Events can move in both directions, with any of the layers as source or sink.

limited. Data movement requires bandwidth and storage to be available between the layers, and latency may be a concern as well. These sizes may range from Kilobytes for configuration files to Gigabytes to move high-fidelity observations to data archives [31]. Events are a special category as they are by definition light-weight, and they serve as control signals that prompt an action. They typically require low latency transfer from source to sink, and the ability for the receiving layer to perform the action [27].

There are four major *logical mobility models* for the source, the sink, and their direction for this movement, as shown in Fig. 4. The first shows the logical entity move from Edge to Fog to Cloud, such as moving observation streams from Edge to Cloud for archival, with the Fog performing filtering or pre-processing [7]. The second shows the entity move from Cloud to Edge, for e.g., flashing a new device image into a Pi deployed in the field, with the image cached in the Fog. Unlike these two “open loop” models, IoT applications often have a “closed loop” that consists of Observe, Decide and Act, and the last two modes indicate these feedback cycles. For e.g., the data source can be on the Edge, decisions are made at the Fog and/or Cloud, and the notification arrives back at the Edge for a response [7,30]. Similarly, the Cloud could provide a data stream to the Fog or Edge that respond back with a control message to enact a service call in the Cloud. Of course, subsets of these paths may be relevant for an application as well, e.g., start at the Fog, move to Cloud and then to Edge.

3.2 Quality of Service (QoS) Characteristics

There are several application QoS requirements that can drive the mix of resource layers to be considered. *Latency and bandwidth* are two obvious factors for applications, and [13] has earlier grouped applications based on their ability to withstand such delay. *Inelastic applications* require real-time processing – typically related to safety of humans or equipment such as autonomous vehicles, or near real-time processing for, say, applications consuming transient

sensor streams [6]. The latency accumulates across each layer that the data passes through, and the Fog may offer a better pair-wise cumulative latency and bandwidth with the Edge and the Cloud, compared to directly between the two. *Elastic applications* are delay-tolerant and designed for off-line or batch processing, such as surveys using drones for planning [14].

The *robustness* of the application becomes important as well. Mission-critical applications that operate City infrastructure such as power grids or traffic signaling may require deterministic network connectivity and computing capacity even during disasters, that may be better provided close to the Edge or Fog [27, 30]. On the other hand, if the robustness relies on instantaneous availability of large computing capacity or scalable services, say for supporting periodic events at stadiums, Clouds offer this flexibility provided redundancy of communication channels and data centers are built in.

Lastly, *cost and user base* can play a major factor in the use of various resources. For applications that have a growing user community, scaling based on an elastic service-based model offered in the Cloud or the Fog allows them to pay proportional to their community size and revenue [5, 19]. Sometimes, applications coupled to consumer hardware, like autonomous cars and wearables, come with in-built (free) computing capability, on the device or a smart phone, that can (or should) be leveraged. Others like city utilities with a slow-changing population may wish to have captive resources self-managed at the Edge and Fog due to the scale that can be leveraged, and complemented by the Cloud [30]. Sometimes, the resource availability itself may increase based on the demand from these applications, such as Cloud data centers or cell phone towers coming up in regions of revenue growth.

3.3 Policies and Constraints

Privacy and security is a growing concern as the number of connected systems grow. There may be explicit corporate policies or national laws that do not allow applications to move data or services outside network or geographical boundaries [30]. Here, the Edge and the Fog may be critical to meet performance needs as also legal compliance. On the other hand, Cloud data centers in general have stronger physical and digital security systems in place that may surpass what can be provided for widely distributed Edge and Fog devices that may operate in hostile environments [21].

IoT applications are often designed as *Cyber Physical Social Systems (CPSS)*, that span data and software (cyber), infrastructure and devices (physical), and human interaction (social). In some cases, the application is intrinsically tied to specific devices (e.g., drones) [14]. In others, they may have explicit dependencies on computing infrastructure or resource capabilities. For e.g., a video classification may require model-training to take place on GPUs or FPGAs, which may be available only on the Edge or Fog. The application may also be coupled to a particular user who is mobile, and it will need to use logical mobility to move state and services to the resource that is physically closest to the user [5]. Consequently, some of the applications may be tightly-coupled or

constrained to those cyber and physical systems.

3.4 Application Use Cases

3.4.1 Urban Surveillance

There is a global push toward Smart Cities and India is poised to upgrade over 100 cities as part of a national initiative ⁴ [17]. As part of a *Light-Pole Computing* pilot, several city blocks in Bangalore are installing video cameras, environmental sensors, Edge computing platforms (e.g., Raspberry Pi) and, at each city block, accelerated Fog computing platforms (e.g., NVIDIA Jetson TX1) ⁵.

Given the advances in deep learning, video surveillance in urban environments is essential not just for public safety but also as a proxy for ambient observations. For e.g., these analytics can be used to automatically identify parking violations, classify vehicles and people, generate visual summaries for consumption by safety agencies, and even blur regions for privacy [13,31]. Training the neural network models for classification is computationally costly, and the source video streams at the edge have large sizes as well. Training may be required as frequently as every few hours to adapt to local conditions. Classification using trained models is faster, but even GPU accelerated Fog platforms can process only hundreds of frames per second.

Here, the training can be done in batch but moving the compute and data to the Fog saves bandwidth. The classification can move to the Edge or Fog, based on latency needs. At night, when women's safety is a concern, real-time needs trump daytimes use cases of detecting parking violations. As an example of a feedback control loop, the PTZ camera can be controlled by the analytics to automatically zoom into features of vehicles. Safety situations detected at the Edge can trigger an alert to spatially proximate smart phones, or notify officers through the Cloud [11]. The Edge and Fog platforms can also host other IoT applications from public utilities that tap into the video or sensor streams for event-analytics [10].

3.4.2 Smart Power Grid

The *Los Angeles Smart Grid* will serve over 4 *Million* customers in the largest public utility in the US [25]. Net-connected smart meters observe power demand at households and industries and report them periodically back to the utility every few minutes. These run off 2G or P2P communication models. Further, SCADA systems at hundreds of city feeders collect high-frequency data on voltage and current quality across multiple phases, often at KHz rates.

Demand-response (DR) optimization refers to shaping or shifting power demand to match supply capacity [2]. Here, liability and privacy concerns mean

⁴Smart Cities Mission, <http://smartcities.gov.in/>

⁵Smart City test bed, Bharadwaj Amrutur, RBCCPS, 29 November, 2016, <http://www.rbccps.org/smart-city/>

that demand-response and load control decisions happen at multiple-levels. The utility uses global but coarse-grained data to run demand forecasts and determine the curtailment level required [4]. It then notifies the customer gateway, say an Edge device for a consumer or a Fog device for a campus like USC, of the curtailment required and the discount (or penalty) pricing. These gateways then take local decisions to determine curtailment strategies and control, say a smart appliance or an electric car, or centrally change set-points of HVAC systems across campus buildings.

While the DR feedback loop can tolerate delays of several seconds or minutes, another smart grid application is one of *state estimation* to determine the health of the distribution network. Here, computational models run over fine-grained power quality measurements from feeders to detect instability, and take rapid response on the Edge to avoid the whole network from being affected. There are distributed state estimation models that are well suited for Fog devices, with periodic updates pushed to the Cloud and global aggregates pushed back. As one can imagine, cyber-security concerns of such critical infrastructure eliminate the possibility of having multi-tenancy on these Fog resources.

3.4.3 Drones for Asset Monitoring

Lastly, drones offer a novel and emerging platform that highlight the role of mobility in Edge and Fog computing. Swarms of these autonomous platforms are being put to use for asset monitoring for gas pipelines or city infrastructure. Drones have on-board navigation and environmental sensors, visual/IR cameras, and sometimes even actuators (e.g., probes, grapples) [14]. They have wireless tethers with mobile or static base-stations, often on a line-of-sight, that act as the Fog layer. Upon return, these drones are charged by the base station and their data off-loaded before the next sortie. Static base stations will have high-speed connectivity to the Cloud, while mobile ones may rely on 4G.

The drones perform on-board real-time Edge computing for navigation [13], combined with control signals from the Fog, which is required for its survival. Additional processing of observations for applications may depend on the spare computing capacity, battery levels and the elasticity needs of the application. The mobility of the drones and the base allows P2P mechanisms to be effected between the drones using high-speed millimeter wave line-of-sight communication or even between multiple bases. This can be used to move data or processes among them to conserve battery or reduce application latency, such as a mini-drones with constrained resources transmitting data to larger drones while in-flight [14].

4 Fog Computing Platform Abstraction

Fog computing, even as a concept, is evolving and as a result, there is limited insight on what the *application runtime and management platform* will turn out to be. They have the possibility of incorporating a service model similar to the

Cloud as they are not resource-starved like the Edge, but may also inherit some requirements of mobility and performance-sensitivity seen in the Edge [5]. Here, we discuss how the system and application characteristics of Fog computing influence the features and abstractions offered by its platform.

4.1 Platform Management

One of the key gaps in the realization of Fog computing is the lack of a platform ecosystem to design and run applications that can leverage the Fog, in conjunction with Edge and Cloud layers. There are two key parts to this: a *fabric* for device management, and a *platform* for application runtime management. Cloud computing fabrics expose every resource “as a Service”, and this in part is a reason for its success [5, 27]. Fabrics like OpenStack manage, schedule and instantiate VM images and instances, block and table storage, and networking capacity. Virtualization allows applications and Big Data platforms designed for desktop and cluster computing work equally well on Clouds.

Device management and application platforms on the Edge is a challenge. Edge computing is still done in either an *ad hoc* manner, or tightly coupled to a smart phone platform like Android using sandboxed “apps”. There are emerging IoT specifications like IETF CoRE, OASIS MQTT and oneM2M to enable management, data transfer and control of edge and embedded devices. IoT edge management software like Azure IoT Gateway, Amazon AWS Greengrass and VMware Liota are starting to be available. Application platforms like Apache Edgent and MiNiFi attempt to make logic deployment and migration on the Edge easier. However, this is still evolving.

Fog computing is at an even earlier stage of platform maturity, but can gain from advances in both Cloud and Edge. Slimmed-down versions of Cloud fabrics could manage clusters of Fog devices, though they may have to operate in a wide-area network, while Fogs that are resource-light can make use of platform software for the Edge [6]. An IaaS model for Fog resources would offer the most flexibility, but interoperability of APIs becomes even more important compared to Clouds – large-scale deployment of Fog resources will take time, and utilizing resources from different providers, like “roaming” using cell-phone providers, is essential [15]. Containerization like `lxc` and Docker are likely to be more favored for Fog rather than hypervisors, though advances like Ubuntu’s `lxd` may make Linux VMs responsive while also offering capabilities like live-migration which can be invaluable for mobile Fog resources [5, 30].

4.2 Application Composition

The execution environment offered by Edge, Fog and Cloud is inherently distributed, and the application space is vast as well. There are many application definition models used in such scenarios such as control flows, data flows, event-driven models [29], and so on. Directed Acyclic Graphs (DAGs) are popular for capturing flow dependencies in complex distributed applications, and are widely used for Cloud and Edge computing [24]. At the same time, latency sensitive

applications may prefer an event-driven model that react rapidly to changing situations [31].

The ability to encode priorities among tasks and temporal processing guarantees for events will be crucial as well for mission-critical systems. Besides latency, energy and costs are likely to be important parts of the application specification [6,9]. Geo-fencing policies may also limit the movement of data or services to specific resources.

The data flowing between loosely coupled parts of an application may be based on *streams*, *micro-batches* or *files*, again depending on the latency and processing costs. While streams offer a low-latency data transfer for sensor observations, micro-batches have grown popular off-late as they offer a balance between managed latency and higher throughput by amortizing per-tuple overheads⁶. Files are well suited for defining batch processing applications. The ability to define specialized data structures, compression and transport mechanisms for distinct stream types such as audio and video may be necessary as well.

State is likely to play a key role in such applications, and the composition models need to offer a way to include it as a first-class entity. The state may be associated with a user, device or session, and it will need to migrate across space (Edge/Fog devices) and time, and offer a context for execution [5]. Another unique element of such application is the role of *spatial proximity* or location-awareness in determining actions. For e.g., proximity of two devices may trigger an action, such as a data transfer, and this may be dictated by their geo-location as well [14]. Some may wish to explicitly couple the application definition with the placement of their constituent tasks, either for performance reasons (e.g., sensing task on edge, processing task on Fog) or for spatial locality to Edge [11].

4.3 Orchestration and Coordination

The application definition needs to be scheduled and coordinated in order to meet various QoS goals such as latency, energy and monetary constraints. This coordination can be done using different strategies. Three common orchestration models that are relevant in such a multi-layered and distributed resource environment are *centralized*, *hierarchical* and *peer-to-peer (P2P)*. We also distinguish between *scheduling decisions*, the *flow of control signals* and the *flow of data*, and different coordination models could be applied to these.

Centralized orchestration has a single service, either per application or for the platform, that is located in one of the three resource layers, and is responsible for making scheduling decisions, and coordinating the transfer of control signals and/or data items. For e.g., in [9], a central resource management layer determines the best resources to schedule the incoming tasks using the monitoring information from the Edge, Fog and Cloud resources. This is simple to design but can suffer from high latencies and transfer costs, and is a single point of failure. While this orchestrator often runs in the Cloud (to coordinate across

⁶Apache Flink, Apache Spark Streaming

edge devices) or the Edge (to interact with different Cloud services), the Fog layer could offer a sweet-spot for such a centralized coordinator. For e.g., when a Fog sits at the gateway between private and public networks, it is in the sole position of being able to judge the behavior of resources in the Edge (private) and the Cloud (public) [1].

A *hierarchical* architecture is a generalization of the centralized model, and allows only vertical communication of data and controls to take place between adjacent layers. This is a natural fit for Fog computing as it leverages both the bandwidth and latency benefits of the Fog layer in accelerating these flows, as well as the compute benefits closer to the observation source [9, 11, 27, 27, 30, 31]. Often, the Cloud forms the root of this tree and is used for global data aggregation and coordination. Local data analytics is delegated to Cloudlets and further to the edge devices. This allows a federated behavior that has shown to scale.

P2P is a form of distributed coordination that avoids a single point of failure. Here, peers in the same Edge or Fog layer can pass control and data directly among each other [28]. The horizontal communication channels may initially be setup by an entity that has a global picture of the resources. This is typically done at the Cloud or the Fog, or one of the edge devices that serves as a leader. There simple component based models for composing and executing P2P applications, as well as complex ones that use Distributed Hash Table (DHT) to maintain an overlay network over peers that frequently enter and leave the system. For example, in [15] the fog devices peer amongst each other for content delivery and service provisioning for improving the performance. Similarly [26] consider a hierarchical architecture where proximate smart traffic lights coordinate among themselves to send green traffic wave or alert the approaching vehicles.

In a *hybrid* model, there are no strict limitations on the flow of control or data flows, and all layers are seen as having resources of heterogeneous characteristics. While there can be interconnections among resources within each layer (Cloud, Fog, Edge), communication can also take place vertically [5]. This can help in different situations. If a higher layer is not reachable due to network failure or resource unavailability, then the tasks can be accomplished within the lower layers itself though with a graceful degradation of the QoS [16]. Similarly, if a Fog layer is unable to reach an edge device, it can replicate the last state of the application running on it to a different edge device. Such hybrid strategies are likely to require more complex coordination, but can potentially improve the resilience of the application.

4.4 Managing Dynamism and Mobility

A Fog computing platform has to be responsive to various forms of *dynamism*. As identified before, there can be resource mobility at the Edge and Fog layers. The applications may also impose requirements on logical mobility of processes and data. Further, there may be changes in the data generation rates, network behavior or energy levels of batteries that requires reactive strategies [24]. Thus,

the runtime environment should offer monitoring capabilities to determine when such adaptation is required, and provide transparent mechanisms for enacting these changes [9]. This may even require *changing the coordination strategy* from, say, centralized to P2P.

Such *migration* may involve moving just state from one service to another, moving an entire application and its dependencies, or moving a VM or container as a whole. For e.g., [5] migrate the user’s data from one Cloudlet to another as per the mobility of the user in order to minimize latency. Depending on whether the Edge and or the Fog resources are physically mobile, we may need to preemptively off-load to a different resource or layer when proximity or connectivity is going to be lost, or periodically save state to the Cloud [24].

For real-time applications or those that have a closed control loop are particularly sensitive to mobility and require careful platform design. Often, these preclude the use of mobile resources, or retain the decision logic in a single resource. For e.g., [18], perform video analytics to blur the scenes of movies based on the mental state of the user with a medical condition, but offload the signal processing task to a static Fog server. Batch applications offer the most flexibility and can efficiently make use of mobile Edge and Fog resources to opportunistically offload the tasks or data as and when resources are available. For e.g., mini drones may be able to offload their data to larger drones or to a Fog server at the base-station when they come within range [14].

5 Related Work

There have been several early papers that conceptualize the idea of Fog computing and Cloudlets. Cisco [6] popularized the notion of Fog computing as a complementary computing layer from Cloud that is driven by distinct applications that require low and predictable latencies. They discuss the importance of multi-tenancy on the fog layer while ensuring mission-critical operation, using a “Foglet” as a software agent on the Fog layer to manage the local operations and interface with the Cloud. We go beyond this, and discriminate between edge devices and the Fog layer, and discuss the interaction between all three layers. We also include mobility as first class entity, not just at the Edge but also at the Fog layer. Further, while a centralized, distributed or hierarchical models are mentioned, we contrast between control flows and data flows that may each use a different interaction model.

[20] introduced the concept of Cloudlets as resource-rich infrastructure that is within one network hop of the mobile edge device, consistent with the concept of Fog Computing, as a second-level data center that is proximate to mobile devices at the edge [22]. They offer examples of the latency benefits for augmented reality and face recognition applications, but these treat Cloudlets more as a “CDN in reverse” to avoid latency and bandwidth costs. They use VMs used to deploy and manage applications, either through active migration or of state overlays over existing VM images. They also discuss the business model for Cloudlets, spanning between retain owners and service providers.

[28] attempt to define Fog computing, but views it from the narrow prism of network management and connectivity. They also tabulate various features and challenges in brief. Our interest in this paper is from the application, platform and middleware perspective. We further consider the system features and the applications that benefit or need Fog.

Some early research investigates platform and application models for Fog computing. [11] propose a programming model for composing applications that run across mobile (Edge), Fog and Cloud layers as a Platform as a Service (PaaS). They offer a multi-way 3-level tree model where the computation is rooted in the Cloud, resources are elastically acquired in the Cloud and Fog layers, and communication is possible between Cloud and Fog, or Fog and Edge. A strictly hierarchical model while simple, limits the flexibility in application composition. Their example applications do not consider a role for the Cloud either, though their APIs support it. This degenerates to a client-server model between the edges and their Fog parent. Further, the interactions between edge devices and Fog layer should be actively used as well, rather than only vertical interactions. Lastly, while mobility of the edge is discussed, this does not consider when the Fog can be mobile as well.

The role of virtualization in enabling Cloud computing is discussed in [5], and they see a similar role for Fog computing as well. They conceive of a VM encapsulating all necessary dependencies for an edge application or user to be hosted on a Cloudlet within 1 hop of the edge, with this VM moving with the edge user to remain at 1-hop distance. This virtualization architecture should expose API for the developers to offload data and processing, synchronization of data among replicas, discovery of Cloudlet resources, and the migration of the VMs across Cloudlets. However, we take a broader platform view and discuss the possible architectural designs for the Fog.

Other literature examine specific applications that benefit from the Fog layer. [8] discuss the role of Fog computing for IoT as Edge computing alone is inadequate to deal with multiple IoT applications. Fog helps with coordination of distributed edge devices and uses Cloud resources. They consider Sense-and-actuate and stream processing as two programming models. But we argue there can be more diverse application composition and coordination models.

[30] discuss the need of Fog computing for real-time applications such as sensing of gas pipelines, smart agriculture and control systems inside factories. They consider a hierarchical architecture where the data is analyzed and processed at one level and then sent to the higher level for further aggregation and analysis. Other possible architectural designs and types of applications are missing. Similarly, [27] also discuss the need of Fog computing for smart city applications. Further they have looked into some of the privacy and security issues that are possible in fog computing. What is lacking in these is a broader examination of application characteristics rather than examples that motivate the Fog, which we address.

[15] attempt a similar effort such as our paper, but limit their exercise to mobile devices rather than edge devices at large. They recognize that the Fog can be static or mobile, similar to entertainment systems in vehicles. They

highlight that Fog can deliver location-aware content unlike Cloud, but this reduces the value of a Fog to that of a CDN, only closer to the edge. However, it is important to note that many Cloud services are capable of using geolocation using network or device GPS to offer location sensitive information. Rather, we state that the physical proximity also offers a certain degree of trust by the client of the Fog, and the ability to host rich interactive services, not just content. Their discussion on research problems delves more on the networking and communication between Mobile/Cloud and Fog, and between Fogs rather than on application and middleware.

[31] offer a brief survey of Fog computing concepts, in which they include both resource poor devices (which we refer to as edge) and resource rich Cloudlets and Cisco’s IOx. They highlight Augmented reality, Content delivery and Mobile Data Analytics as three motivating applications to reduce the latency delays, and reduce bandwidth costs. They do not offer any analysis of Fog computing architecture or platform dimensions. As before, network management using Network Virtualization and SDN appear as technical issues to tackle. They identify the need for QoS, programming APIs, resource provisioning, security and billing as aspects to address.

[26] present a survey and discuss a hierarchical Fog-based architecture. We discuss several alternative architectural and coordination designs rather than a one-size fits all. They too consider IoT applications but fail to present a taxonomy of the application characteristics that can benefit from Fog as we do.

6 Discussion and Conclusions

6.1 Challenges

There are several interesting problems that arise in the context of Fog computing, and addressing these can pave way for the technical feasibility.

- *Programmability:* It should be easy for the application developers to run their applications on the Fog. This will require the interface and the programming model to hide the complexities of Fog computing. Fog components should also allow for application-specific customization and optimizations [8, 28].
- *Predicting demand:* One of the major challenges for deployment of Fog devices is to predict the users’ demand and accordingly determine the quantity of resources required at different locations [15]. Thus it is a challenge to optimally place the fog devices at appropriate locations so as to serve the requests of nearby mobile users.
- *Power and Network Consumption:* Tasks in Fog devices are distributed across multiple locations compared to more centralized Clouds [8]. Due to the distributed nature of computations, captive local power supply and network connectivity will be important. This can be a challenge for developing countries without 24×7 power supply. Task placement may

need to be optimized to minimize the power consumption, or leverage renewables like Solar.

- *Where to push the tasks?* One has to decide how to partition an application and distribute the tasks among the three layers. This depends on the application requirements [31].
- *Security and fault tolerance:* Since Fog resources are geographically distributed and can have multiple service providers, ensuring application and data security is a key challenge [9,27]. Also, one needs to plan how to handle failures of Fog devices. When the nearby Fog devices are inaccessible, users may need to push their applications to the remote Fog or even the Cloud, which may impact the performance of the applications.
- *Fog providers and billing:* Fog computing services may be provided by Internet, Cloud and Cell-phone service providers, retail merchants, or emerging “gig economies” like taxi or hotel aggregators. Pricing and billing remains a challenge in terms of sustaining a commercial ecosystem or value added services [31]. Moreover there are possible billing models such as *consumption based* where the users are billed as per their usage or *subscription based* where the users pay a fixed price monthly and can use the Fog network-wide [5].

6.2 Reality Check

It is still early days for Fog computing [28]. Growth of fog has nowhere near kept up with the growth in edge devices. So the pain-points or gaps of a lack of fog computing have not reached a threshold. These are starting to emerge more in vertically integrated “private” scenarios rather than horizontal, reusable “public” ones, much in the way of Cloud data centers being private to Amazon, Google and Microsoft, before the business model fell into place for commercialization [31].

We do not have large scale deployments of Fog Computing or its commercial operation in a pay-as-you-go on-demand model like public Clouds currently available. But the need for it is growing and we can foresee several existing infrastructure operators evolving to offer Fog services as well. Two prime contenders are *operators of cell-phone towers* who tend to have captive power, communications and space, and can complement this with computing resources [22], and *taxi aggregators* like Uber and Ola who are already offering value added services such as WiFi and grocery delivery, and can incrementally extend this to a Fog server hosted in the boot of their cab. At the same time, emerging infrastructure such as *Smart Cities* will see the organic growth of the Fog layer, initially integrated as part of a vertical domain such as smart power grid or smart transportation (e.g., collocated with a transformer or traffic light, at each city block), and later offered as a horizontal resource shared by multiple city or commercial services [6, 7, 27]. Such “light-pole” sensing and computing is already starting, as mentioned earlier.

At the same time, not all mobile or IoT applications will necessarily need or even benefit from having a Fog Computing layer. For e.g., as Smart Power Meters are rolled out across Los Angeles, the largest public utility in the US, these meters record the electricity usage data at 15 *min* granularity to support demand-response optimization decisions that are taken periodically [25]. Even with $\approx 4M$ customers, this works out to about 400 *GB* of data collected each day (assuming a 1*kb* payload per observation), or about 4 *GB* of distributed data collected per 15*min* interval. This is meager, even for a mega-city like Los Angeles, compared to the 24 *GB/sec* streamed by Netflix during peak hours ⁷. While there are other Smart Grid applications that may require higher sampling rates and data sizes (e.g., realtime analytics over Phasor Measurement Units (PMU) data [12]), these are still evolving. So not every Smart City or IoT application requires low latency or high bandwidth or complex analytics.

The concept of Utility computing itself existed long before Cloud computing in its current form emerged as a feasible business model, with Grid computing having meanwhile been tried, less successfully, in the academic community [16, 32]. Likewise, sensor networks and pervasive computing concepts existed a decade before the technologies converged to offer IoT at massive scales. Likewise, the reality is that the gestation period for Fog computing may be anywhere from a couple of years to a decade before it matures into a sustainable technology and business model. In fact, one of the early analogies to Fog computing was the *Google Search Appliance*, which offered on-premises search capability for enterprises, with the Cloud only for support. But this died a natural death with network bandwidth improving and the Cloud capabilities like deep-learning far out-stripping what was possible at the GSA ⁸. So the eventual outcome of Fog computing may indeed be different from what one might envision, but offering dimensions for characterization allows this eventual manifestation to fall at some point in the spectrum.

References

- [1] M. Aazam and E. N. Huh. Fog computing and smart gateway based communication for cloud of things. In *2014 International Conference on Future Internet of Things and Cloud*, pages 464–470, 2014.
- [2] S. Aman, M. Frincu, C. Chelmiss, M. Noor, Y. Simmhan, and V. K. Prasanna. Prediction models for dynamic demand response: Requirements, challenges, and insights. In *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 338–343, Nov 2015.
- [3] S. Aman, Y. Simmhan, and V. K. Prasanna. Energy management systems: state of the art and emerging trends. *IEEE Communications Magazine*, 51(1):114–119, January 2013.
- [4] S. Aman, Y. Simmhan, and V. K. Prasanna. Holistic measures for evaluating prediction models in smart grids. *IEEE Transactions on Knowledge and Data Engineering*, 27(2), 2015.

⁷Evolution of the Netflix Data Pipeline, February 15, 2016, <http://techblog.netflix.com/2016/02/evolution-of-netflix-data-pipeline.html>

⁸http://www.theregister.co.uk/2016/02/10/google_quits_search_appliances/

- [5] L. F. Bittencourt, M. M. Lopes, I. Petri, and O. F. Rana. Towards virtual machine migration in fog computing. In *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, 2015.
- [6] Flavio Bonomi, Rodolfo Milito, Preethi Natarajan, and Jiang Zhu. *Fog Computing: A Platform for Internet of Things and Analytics*, pages 169–186. Springer International Publishing, Cham, 2014.
- [7] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *MCC workshop on Mobile cloud computing*. ACM, 2012.
- [8] A. V. Dastjerdi and R. Buyya. Fog computing: Helping the internet of things realize its potential. *Computer*, 49(8):112–116, Aug 2016.
- [9] Amir Vahid Dastjerdi, Harshit Gupta, Rodrigo N. Calheiros, Soumya K. Ghosh, and Rajkumar Buyya. Fog computing: Principles, architectures, and applications. *CoRR*, abs/1601.02752, 2016.
- [10] Rajrup Ghosh and Yogesh Simmhan. Distributed scheduling of event analytics across edge and cloud. *CoRR*, abs/1608.01537, 2016.
- [11] Kirak Hong, David Lillethun, Umakishore Ramachandran, Beate Ottenwälder, and Boris Koldehofe. Mobile fog: A programming model for large-scale applications on the internet of things. In *ACM SIGCOMM Workshop on Mobile Cloud Computing, MCC '13*, pages 15–20. ACM, 2013.
- [12] John Interrante and Kareem S Aggour. Applying cluster computing to enable a large-scale smart grid stability monitoring application. In *High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESSE), 2012 IEEE 14th International Conference on*, pages 328–335. IEEE, 2012.
- [13] Jiong Jin, Jayavardhana Gubbi, Tie Luo, and Marimuthu Palaniswami. Network architecture and qos issues in the internet of things for a smart city. In *IEEE International Symposium on Communications and Information Technologies (ISCIT)*, 2012.
- [14] Seng Wai Loke. The internet of flying-things: Opportunities and challenges with airborne fog computing and mobile cloud in the clouds. *CoRR*, abs/1507.04492, 2015.
- [15] Tom H. Luan, Longxiang Gao, Zhi Li, Yang Xiang, and Limin Sun. Fog computing: Focusing on mobile users at the edge. *CoRR*, abs/1502.01815, 2015.
- [16] Henrik Madsen, Bernard Burtzsch, G Albeanu, and FL Popentiu-Vladicescu. Reliability in the utility computing era: Towards reliable fog computing. In *IEEE International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2013.
- [17] Prasant Misra, Yogesh Simmhan, and Jay Warrior. Towards a practical architecture for india centric internet of things. *CoRR*, abs/1407.0434, 2014.
- [18] K. Sadeghi, A. Banerjee, J. Sohankar, and S.K.S. Gupta. Optimization of brain mobile interface applications using iot. In *IEEE International Conference on High Performance Computing*, 2016.
- [19] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4):14–23, Oct 2009.
- [20] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. The case for vm-based cloudlets in mobile computing. *IEEE pervasive Computing*, 8(4):14–23, 2009.
- [21] Mahadev Satyanarayanan, Grace Lewis, Edwin Morris, Soumya Simanta, Jeff Boleng, and Kiryong Ha. The role of cloudlets in hostile environments. *IEEE Pervasive Computing*, 12(4):40–49, 2013.
- [22] Mahadev Satyanarayanan, Rolf Schuster, Maria Ebling, Gerhard Fettweis, Hannu Flinck, Kaustubh Joshi, and Krishan Sabnani. An open ecosystem for mobile-cloud convergence. *IEEE Comm. Magazine*, 53(3):63–70, 2015.

- [23] Mahadev Satyanarayanan, Pieter Simoens, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, Wenlu Hu, and Brandon Amos. Edge analytics in the internet of things. *IEEE Pervasive Computing*, 14(2):24–31, 2015.
- [24] Cong Shi, Vasileios Lakafosis, Mostafa H Ammar, and Ellen W Zegura. Serendipity: enabling remote computing among intermittently connected mobile devices. In *ACM international symposium on Mobile Ad Hoc Networking and Computing*, pages 145–154. ACM, 2012.
- [25] Yogesh Simmhan, Saima Aman, Alok Kumbhare, Rongyang Liu, Sam Stevens, Qunzhi Zhou, and Viktor Prasanna. Cloud-based software platform for data-driven smart grid management. *IEEE/AIP Computing in Science and Engineering*, July/August:1–11, 2013.
- [26] I. Stojmenovic. Fog computing: A cloud to the ground support for smart things and machine-to-machine networks. In *2014 Australasian Telecommunication Networks and Applications Conference (ATNAC)*, pages 117–122, 2014.
- [27] I. Stojmenovic and S. Wen. The fog computing paradigm: Scenarios and security issues. In *Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2014.
- [28] Luis M Vaquero and Luis Roderio-Merino. Finding your way in the fog: Towards a comprehensive definition of fog computing. *ACM SIGCOMM Computer Comm. Review*, 44(5):27–32, 2014.
- [29] Matt Welsh, David Culler, and Eric Brewer. Seda: An architecture for well-conditioned, scalable internet services. *SIGOPS Operating Systems Review*, 35(5):230–243, October 2001.
- [30] M. Yannuzzi, R. Milito, R. Serral-Graci, D. Montero, and M. Nemirovsky. Key ingredients in an iot recipe: Fog computing, cloud computing, and more fog computing. In *IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2014.
- [31] Shanhe Yi, Cheng Li, and Qun Li. A survey of fog computing: concepts, applications and issues. In *ACM Workshop on Mobile Big Data*, 2015.
- [32] L. Youseff, M. Butrico, and D. Da Silva. Toward a unified ontology of cloud computing. In *2008 Grid Computing Environments Workshop*, pages 1–10, 2008.