Journal of Network and Computer Applications 82 (2017) 152–165

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications

Hamid Reza Arkian[a,*], Abolfazl Diyanat[b], Atefe Pourkhalili[a]

[a] Department of Computer Engineering, University of Guilan, Rasht, Iran
[b] Department of Electrical and Computer Engineering, University of Tehran, Tehran, Iran

ARTICLE INFO

Keywords:
Internet of things (IoT)
Fog computing
Crowdsensing
Resource allocation
Optimization

ABSTRACT

Development of Internet of things (IoT) has revitalized the feature scale of wearables and smart home/city applications. The landscape of these applications encountering big data needs to be replotted on cloud instead of solely relying on limited storage and computational resources of small devices. However, with the rapid increase in the number of Internet-connected devices, the increased demand of real-time, low-latency services is proving to be challenging for the traditional cloud computing framework. Although, fog computing, by providing elastic resources and services to end users at the edge of network, emerges as a promising solution, but the upsurge of novel social applications such as crowd sensing has fostered the need for scalable cost-efficient platforms that can enable distributed data analytics, while optimizing the allocation of resources and minimizing the response time. Following the existing trends, we are motivated to propose a fog computing based scheme, called MIST (i.e. a cloud near the earth's surface with lesser density than fog), to support crowd sensing applications in the context of IoT. For cost-efficient provisioning limited resources, we also jointly investigate data consumer association, task distribution, and virtual machine placement towards MIST. We first formulate the problem into a mixed-integer non-linear program (MINLP) and then linearise it into a mixed integer linear program (MILP). A comprehensive evaluation of MIST is performed by consideration of real world parameters of the Tehran province, the capital of Iran. Results show that as the number of applications demanding real-time service increases, the MIST fog-based scheme outperforms traditional cloud computing.

1. Introduction

Rapid development of computational resources will enable the sensing, capturing, collection, and processing of real-time data from billions of connected devices, and can be envisaged to serve many different applications including wearable computing, smart metering, smart home/city, connected vehicles and large-scale wireless sensor network (Scuotto et al., 2016). It makes everything connected and smarter, and termed the Internet of things (IoT) (Chiang et al.).

IoT is expected to generate large volumes of sensor data (Qin et al., 2016). Due to popularity of Big Data technologies, processing these large volumes of data has become easier than ever. However, still there are some challenges. On one hand, smart devices faced challenges rooted from computation power, battery, storage and bandwidth, which in return hinder quality of services (QoS) and user experience. To alleviate the burden of limited resources on smart devices, cloud computing is considered as a promising computing paradigm, which can deliver services to end users in terms of infrastructure, platform and software, and supply applications with elastic resources at low cost (Botta et al., 2016). With the popularity of utility-based cloud computing, the tendency to collect a large amount of data has been increasing over the last few years (Ahmad et al., 2016).

On the other hand, Big Data today is currently characterized along three dimensions: volume, velocity, and variety. But, many IoT use cases, including smart cities, smart grids, and smart transportation, are naturally distributed (Arkian et al., 2014). This observation suggests adding a fourth dimension to the characterization of Big Data, namely, geo-distribution. Hence, there is a need for distributed intelligent platform at the edge that manages distributed computing, networking, and storage resources (Bonomi et al., 2014b).

The latest trend of computing paradigm is to bring resources such as computation and storage close to data generating entities at the edge of networks (Ahmad et al., 2016), which motivates the promising computing paradigm of fog computing as a result of the prevalence of ubiquitously connected smart devices relying on cloud services (Bonomi et al., 2014b). Fog computing keeps data and computation close to end users at the edge of network, and thus provides a new breed of applications and services to end users with low latency, high

* Corresponding author.
E-mail addresses: arkian@msc.guilan.ac.ir, hamid.arkian@gmail.com (H.R. Arkian).

http://dx.doi.org/10.1016/j.jnca.2017.01.012
Received 6 August 2016; Received in revised form 8 December 2016; Accepted 15 January 2017
Available online 18 January 2017
1084-8045/

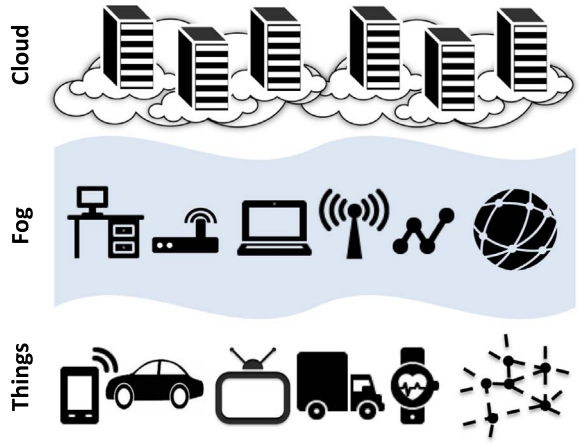

Fig. 1. Cooperating cloud and fog in a three layer service delivery model.

bandwidth, and geo-distributed (Bonomi et al., 2014a). Fog computing is usually cooperated with cloud computing. As a result, things, fog and cloud together form a three layer service delivery model (as depicted in Fig. 1).

In the era of Big Data, fog computing can support edge analytics and stream mining, which can process and reduce data volume at a very early stage, thus cut down delay and save bandwidth (Yi et al., 2015). A good application of the fog for IoT big data is mobile crowd sensing (MCS) that is based on crowd sensing technologies. Crowd sensing technologies have been widely used to collect sensor data in different contexts in IoT paradigm (Jayaraman et al., 2015). In crowd sensing, the focus has been on monitoring large-scale phenomena that cannot be measured using information from a single individual. In an MCS, sensor data (e.g. noise level in an outdoor environment) can be collected from user's mobile device (Ganti et al., 2011). The collected data can be used by applications in the cloud in their decision making process (e.g. determine the noise pollution level at an intersection in the city by fusing data from multiple devices).

1.1. Motivation

As discussed above, there is a need for an efficient architecture that can satisfy the aforementioned requirements to facilitate easy development of IoT mobile crowd sensing applications in the fog. Also, researchers tend to perform well and produce accurate results when filtered relevant data are provided. The most widely needed filtering conditions in the mobile crowd sensing applications are location awareness (i.e. spatial), activity awareness, time awareness (i.e. temporal), and energy awareness (Perera et al., 2015a). Besides, cloud provisioning and resource management are still interesting topics in fog computing environment (Endo et al., 2011). Resource provisioning is the systematic approach of allocating available resources to the needed clients over the network (Arkian et al., 2015). These resources should be allocated optimally to the applications which are running in the virtualized environment. But, selecting suitable solutions from a set is not a trivial task due to the dynamicity, geo-distribution, and all the other different requirements relevant to the fog environment.

Based on the aforementioned problems we highlight the importance of addressing three research challenges:

1. The importance of using fog (i.e. processing data locally before transmitting to the cloud).
2. The importance of having context-aware platform (i.e. filtering data and conditions in mobile crowd sensing platform and producing accurate results).
3. The importance of optimum utilizing devices with low level computational and storage capabilities (i.e. efficient resource provisioning).

1.2. Our contribution

In this paper, we propose a fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowd sensing applications which we call it MIST (i.e. a cloud near the earth's surface with lesser density than fog). The main contributions of this research are listed below.

- First, we propose a new fog-based communication scheme for IoT with a new platform structure which is customized for fog-based mobile crowd sensing in smart city applications of IoT.
- Then, we study a cost efficient resource provisioning problem with guaranteed QoS in the fog architecture. We formulate the cost minimization problem in a form of mixed-integer non-linear programming (MINLP) with joint consideration of data consumer association, task distribution, and virtual machine (VM) placement.
- To deal with the high computational complexity of solving MINLP, we also linearise it as a mixed-integer linear programming (MILP) problem.
- Finally, we consider a simulation of the essential nodes (i.e. data generators, data consumers, fog nodes, and data centres) in the Tehran province as the highest populated province of Iran, the respective population of people using Internet services, and the corresponding geographic location of the data centres. Then, we perform a comparative performance evaluation of cloud computing with that of fog computing for a city with a high number of Internet-connected devices demanding different services.

1.3. Organization

The remainder of the paper is organized as follows. In Section 2, we present some background material and review related literature. The proposed scheme and its components are discussed in Section 3. Section 4 shows the resources provisioning optimization approach proposed in cost-efficient manner. In Section 5, we evaluate the performance of the proposed fog-based scheme in comparison with the conventional cloud through simulation. Finally, Section 6 concludes the paper. Linearisation of the MINLP problem that discussed in Section 4, are presented in Appendix A.

2. Related works

2.1. Internet of things

Recent research has spawned the concept of IoT that connects billions of things across the globe to the Internet and enables machine to machine (M2M) communication among these devices (Diaz et al., 2016). IoT framework is a dynamic and persuasive platform for data storage, computation, and management (Cavalcante et al., 2016).

Different IoT solutions have been proposed in the literature (Perera et al., 2015b). Most of them are focused on the smart home/city domains. A number of generic platforms are being developed (e.g. Ninja Blocks (Schappi, 2013)), and SmartThings (P.G. Corporation, 2013)) to support applications in the domains of the smart home and the smart city.

Cities also have large concentrations of resources and facilities (Scuotto et al., 2016). ParkSight (I. Streetline, 2014) is a parking management technology designed for smart cities. Context information is retrieved through sensors (i.e. magnetometers) embedded in parking slots. Streetbump (Bits, 2013) is a crowd-sourcing project that helps residents to improve their neighbourhood streets. Volunteers use the Streetbump mobile application to collect road condition data while they drive. The data are visualized on a map to alert residents regarding real-time road conditions. The collected data provide governments with real-time information with which to fix problems and plan long-term investments.

2.2. Cloud computing

To provision the resource management and heavy computational potential to the applications, IoT leans highly on cloud computing (Botta et al., 2016). Over the last few years, a good number of works on cloud computing illustrate the detailed underlying process behind the provisioning of cloud services (Cavalcante et al., 2016). Today, cloud computing is a technology that can be defined as a tool which provides enormous benefits to their end users (Agarwal et al.). It is an on-demand service model which is remotely available to users, highly scalable and allocates resources to the users. Cloud computing offers various services as on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service. Cloud computing has a number of benefits such as reduced cost, increased storage, flexibility, reduced time for implementation, and shortened time life cycle (Agarwal et al.).

2.3. Fog computing

First time, Cisco proposed the revolutionary concept of fog computing in Bonomi et al. (2014b). The fog vision was conceived to address applications and services that do not fit well the paradigm of the cloud (Bonomi et al., 2014b). They include: (1) applications that require very low and predictable latency (e.g. video conferencing), (2) geo-distributed applications (e.g. pipeline monitoring), (3) fast mobile applications (e.g. vehicular communication), and (4) large-scale distributed control systems (e.g. smart traffic light systems).

Sarkar et al. (2015) assess the applicability of the fog computing paradigm to serve the demands of different applications in the context of IoT. Tang et al. (2015) proposed a hierarchical distributed fog computing architecture for Big data analysis in the smart cities. In Ahmad et al. (2016), a novel fog-based framework for health and wellness applications was introduced called HealthFog. Hong et al. (2013) designed a programming model to support large-scale IoT applications through mobile fog computing. The model supports the service provisioning to geographically scattered, latency-sensitive applications. Recent researches, however, have revealed some of the important aspects of fog computing. The importance and applicability of fog computing was assessed by Preden et al. (2015) at a superficial level. Do et al. (2015) and Aazam and Huh (2015) have inspected the different intricacies of resource allocation in a fog computing framework.

2.3.1. Fog nodes

Fog nodes are heterogeneous in nature. They range from high-end servers, edge routers, access points, set top boxes, and even end devices such as vehicles, sensors, mobile phones, etc. (Yi et al., 2015). ParaDrop (Willis et al., 2014) is a new fog computing architecture on gateway (i.e. WiFi access point or home set-top box), which is an ideal fog node choice due to its capabilities to provide service and its proximity at the network edge. Cloudlet (Satyanarayanan et al., 2015), like a second-class data centre, is able to provide elastic resources to nearby mobile devices, with low latency and large bandwidth. Zhu et al. (2013) have provided dynamic customizable optimization for web applications based on client devices and local network conditions collected by fog nodes.

2.4. Resource provisioning

Resource selection may be done using optimization algorithms. Many optimization strategies may be used, from simple and well-known techniques such as simple heuristics with thresholds or linear programming to newer, more complex ones, such as Lyapunov optimization (Urgaonkar et al., 2010). Agarwal et al. (2010) define a system called Volley to automatically migrate data across geo-distributed data centres. This solution uses an iterative optimization algorithm based on weighted spherical means. Also, Gu et al. (2015) to solve the problem of the unstable and long-delay links between the cloud data centre and medical devices in the medical cyber-physical systems (MCPSs), integrate fog computation and MCPS and build fog computing supported MCPS (FC-MCPS).

3. Fog-based mobile crowd sensing (MCS) scheme

As identified in the previous sections, there is a need for an efficient architecture to facilitate easy development of IoT crowd sensing applications in the fog. Also, researchers tend to perform well and produce accurate results when filtered relevant data are provided. The most widely needed filtering conditions in the mobile crowd sensing applications are location awareness (i.e. spatial), activity awareness, time awareness (i.e. temporal), and energy awareness (Perera et al., 2015a). Therefore, in this section, we first will discuss about a use case for IoT crowd sensing applications, and then we will propose a customized fog architecture and its corresponding platform to facilitate all of these conditions. The proposed scheme has the potential to form the basis for crowd sensing in smart city applications, which have become very popular and have a clear commercial advantage.

3.1. A use case for IoT crowd sensing applications

An example scenario of smart city is *monitoring citizen activity* (as depicted in Fig. 2). The aim of such an application in a smart city environment is to determine the activity of users in an outdoor park. In this scenario, the key requirements include: (1) ability to perform a task on demand, in this example, capturing data within a given location from users who are involved in certain activities; (2) ability to perform local data analytics, in this example, activity recognition on-the-fly; (3) ability to store analysed data locally in the fog that can be queried later by the cloud (for applications that does not require instantaneous insights); and (4) ability to use an energy-efficient strategy for continuous monitoring and data upload to the cloud. This scenario is a typical example of a mobile crowd sensing application and can be extended to many such crowd sensing-based smart city application scenarios, such as monitoring air pollution by mounting sensing component on buses, cars, trams, or mobile phone; and determining noise pollution in subways using an individual's mobile smart phone as a sound meter, etc. (Hasenfratz et al., 2012).

In the following, our intent is to propose a fog-based scheme that addresses all, or at least the vast majority of the IoT requirements in these applications.

3.2. Customized fog architecture

In this section, we present a high-level overview of our customized architecture. The architecture is designed in a layered model (as depicted in Fig. 3) comprising data generators (DG), fog computing (FC), cloud computing (CC) and data consumers (DC) layers that will be introduced in the following.

3.2.1. Data generators layer

In the customized architecture, layer1 is the edge of network and called as data generators (DG) layer. DG Layer is the sensing network, which contains numerous non-invasive, highly reliable, and low cost sensory nodes, smart mobile devices capturing situational context information from the user and his/her environment, microsized processing platforms (e.g. raspberry pi mounted on buses) and any other Internet connected device. Those entities can be widely distributed at various public infrastructures to monitor their condition changes over time. Note that massive sensing data streams are generated from these sensors that are geo-spatially distributed, which have to be processed as a coherent whole. DG Layer filters the data to be consumed locally (e.g. by actuators), and sends the rest to the higher

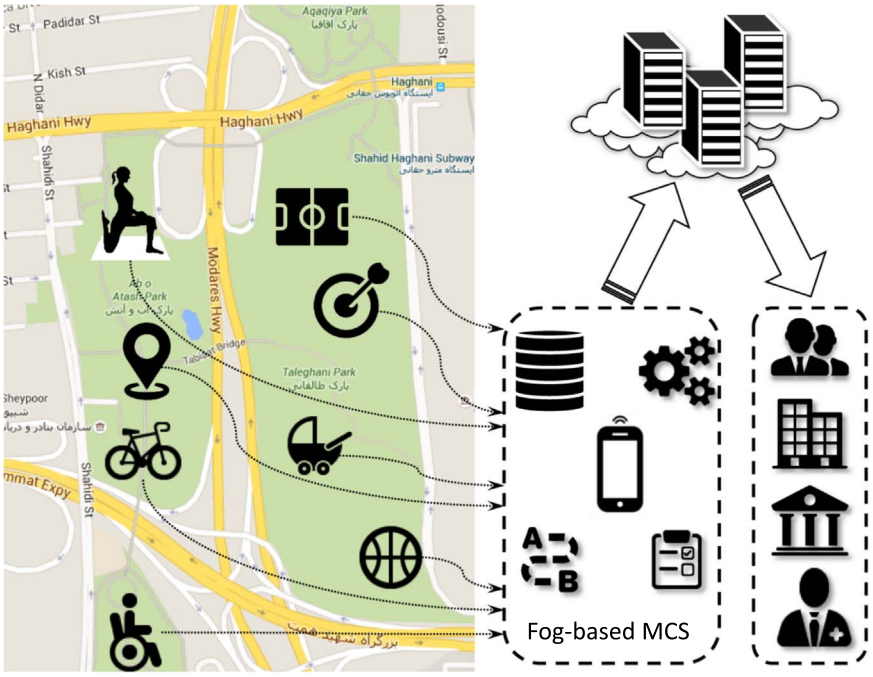

Fig. 2. Monitoring citizen activities in smart cities. The aim of such an application is to determine the activity of users in an outdoor park.

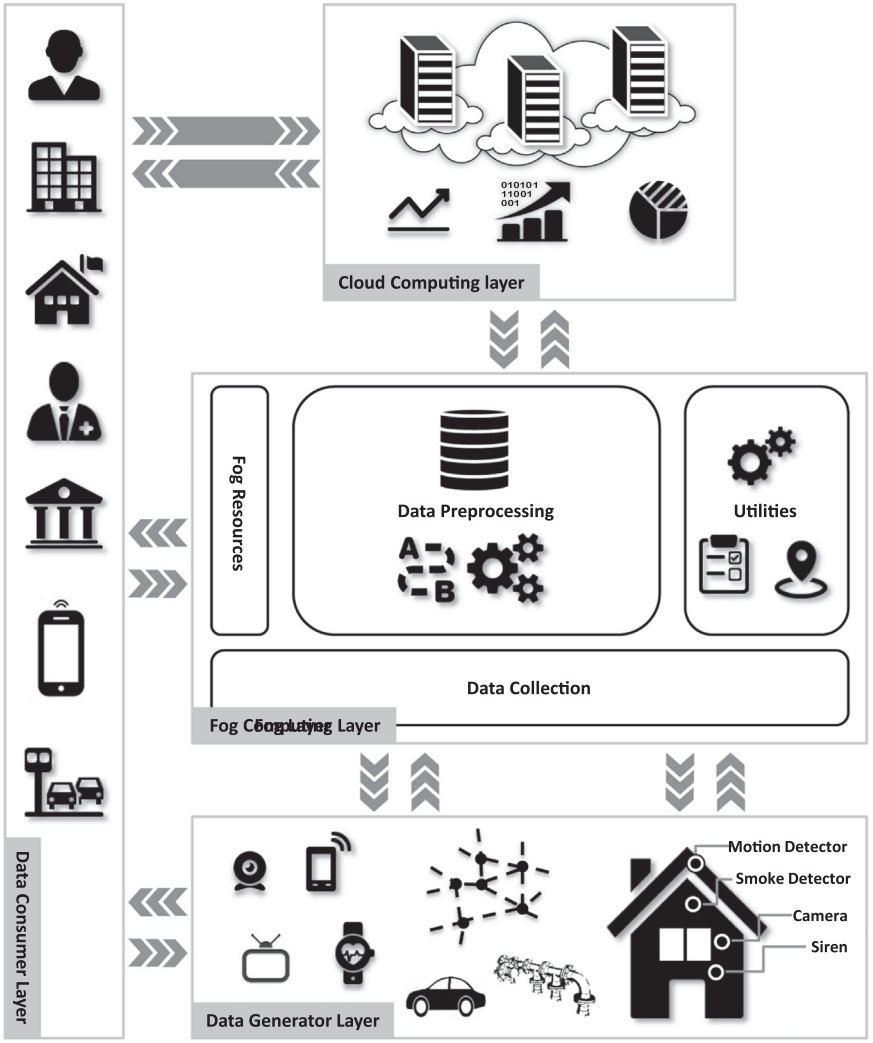

Fig. 3. Proposed four layer fog architecture. Data consumers can submit their requests to the other three layers and get corresponding services.

layers. The captured data can also be augmented with situation information like time, temperature, date. Note that context awareness allows to eliminate a significant amount of uninterested data from communicating over limited network resources. Hence, the data are expected to be collected based on context information and conditions provided by the data consumers.

3.2.2. Fog computing layer

The nodes at the edge (Layer1) forward the raw data into the next layer, fog computing (FC) layer, which is composed of many low-power and high-performance computing nodes or smart mobile devices. Each node is connected to and responsible for a local group of sensors that usually cover a neighbourhood or a small community, performing data analysis in a timely manner. The output of the fog layer has two parts: the first are reports of the results of data processing to the upper layer for large-scaled and long-term behaviour analysis and condition monitoring, while the second is simple and quick feedback control to the local data consumers. For example, in a smart city, if one segment of the gas pipeline is experiencing a leakage or a fire, these computing nodes will detect the threat and call control centre to shutdown the gas supply to this area.

3.2.3. Cloud computing layer

While fog provides localization, therefore enabling low latency and context awareness, the cloud provides global centralization. Many applications require both fog localization and cloud globalization, particularly for Big Data analytics.

The top layer in the customized architecture is a cloud computing layer, providing city-wide monitoring and centralized control. Complex, long-term, and city-wide behaviour analyses can also be performed at this layer, such as large-scale event detection, long-term pattern recognition, and relationship modelling, to support dynamic decision making. This allows municipalities to perform city-wide response and resource management in the case of a natural disaster or a large-scale service interruption.

3.2.4. Data consumers layer

Wide range of data consumers can be considered for the IoT services, from actuators and individual users to insurance companies, educational institutes and to city council and research organizations. As illustrated in Fig. 3, the data consumers layer is perpendicular to the other three layers. The reason is that data consumers can submit their requests to the other three layers and get corresponding services.

3.3. Fog-based MCS platform

In this section we discuss about the components of the proposed fog-based MCS platform (as presented in Fig. 4), and explain in detail how the objectives are met. Note that the proposed platform leverages the strengths of some related researches namely context-aware open mobile miner (CAROMM) (Sherchan et al., 2012), on-demand mobile distributed sensing platform (C-MOSDEN) (Perera et al., 2015a) and scalable energy-efficient distributed data analytics platform (CARDAP) (Jayaraman et al., 2015).

3.3.1. Fog resources tier

As discussed earlier, fog nodes are heterogeneous in nature. They range from high-end servers, edge routers, access points, set-top boxes, and even end devices such as vehicles, sensors, mobile phones, etc. The different hardware platforms have varying levels of RAM, secondary storage, and real estate to support new functionalities. The platforms run various kinds of OSes and software applications, resulting in a wide variety of hardware and software capabilities. The fog network infrastructure is also heterogeneous in nature, ranging from high-speed links connecting enterprise data centres and the core to multiple wireless access technologies (e.g. 3G, 4G, WiFi, etc.) towards the edge.

3.3.2. Fog abstraction tier

The fog abstraction layer hides the platform heterogeneity and exposes a uniform and programmable interface for seamless resource management and control. The tier provides generic APIs for monitoring, provisioning and controlling physical resources (i.e. storage, compute and networking resources). Moreover, the tier includes necessary techniques that support virtualization, specifically the ability to run multiple OSes or service containers on a physical machine to improve resource utilization. Virtualization enables the abstraction tier to support multi-tenancy. The tier also exposes generic APIs to specify security, privacy and isolation policies for OSes or containers belonging to different tenants on the same physical machine.

3.3.3. Fog service management tier

The service management tier provides dynamic, policy-based life-cycle management of fog services. The Management functionality is as distributed as the underlying fog infrastructure and services. Managing services on a large volume of fog nodes with a wide range of capabilities is achieved with the following technology and components.

- *Foglet*: The Foglet agent uses abstraction tier APIs to monitor the health and state associated with the physical machine and services deployed on the machine. This information is both locally analysed and also pushed to the distributed storage for global processing. Foglet is also responsible for performing life-cycle management activities such as standing up/down guest OSes, service containers, and provisioning and tearing down service instances, etc.
- *Distributed database*: A distributed database, while complex to implement is ideal for increasing fog's scalability and fault-tolerance. The distributed database provides faster (than centralized) storage and retrieval of data. The database is used to store both application data and necessary meta-data (e.g. fog node's hardware and software capabilities, health information, etc.) to aid in fog service management.
- *Policy manager module*: The service management tier provides policy-based service routing, i.e., routes an incoming service request to the appropriate service instance that confirms to the relevant business policies. The framework achieves this with the help of the policy manager module. Administrators interact with the service management tier via an intuitive dashboard-style user interface (UI). Administrators enter different policies (e.g. QoS, service configuration, security, etc.), manage, and monitor the fog platform through this UI.

3.3.4. Applications APIs

The fog software framework exposes some APIs that applications use to effectively leverage the fog platform. These APIs are broadly classified into data and control APIs. Data APIs allow an application to leverage the fog distributed data store. Control APIs allow an application to specify how the application should be deployed on the fog platform.

3.3.5. Data collector module

The Data Collector Module of the proposed platform uses plugins and virtual sensors to interface with data generators. The plugin and virtual sensors together enable integration of heterogeneous data sources to the platform. The virtual sensor is an abstract representation of a physical/logical sensor in the proposed system.

3.3.6. Activity-aware module

The activity-aware module is capable of recognizing different activities. Users can combine these activities to build different types of queries. STAR framework (Abdallah et al., 2012) (stands for stream learning for mobile activity recognition) is a good example. STAR is an adaptive framework for activity recognition that incrementally learns from evolving data stream.

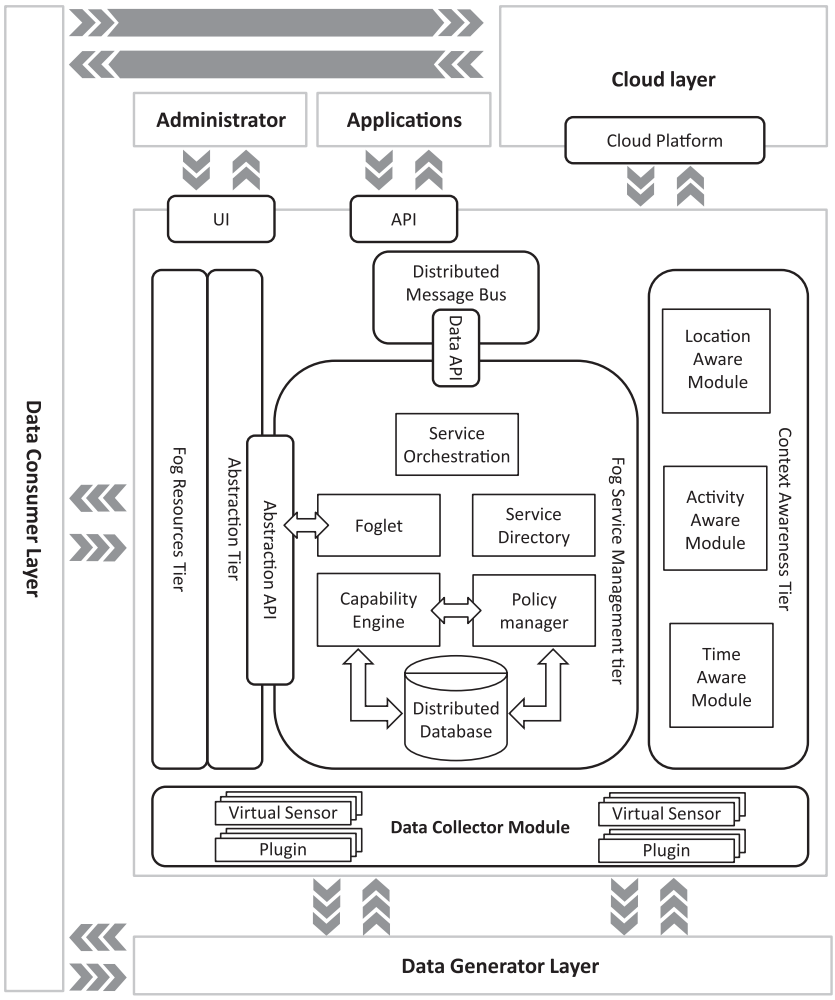

Fig. 4. Proposed fog-based MCS platform.

3.3.7. Location-aware module

The location-aware module is capable of recognizing when the device moves into a certain area and moves away from a certain area. Location awareness can also be combined with other sensing parameters. Sensor data retrieved by different plugins can be referred in different queries.

4. Cost-efficient resource provisioning optimization approach

As mentioned in the previous section, in order to fully manifest the distributed architecture, the fog platform leverages a provisioning mechanism for scalable and automatic resource management. The resources may classes:

- Computing, requiring the selection of hyper-visors in order to virtualize both the computing and I/O resources.
- Storage, requiring a virtual file system and a virtual block and/or object store.
- Networking, requiring the appropriate network virtualization infrastructure (e.g. software defined networking technology).

Generally, there are four fundamental challenges inherent to resource allocation that need special emphasis in fog-based platforms (Endo et al., 2011): resource modelling, resource offering and treatment, resource discovery and monitoring and also resource selection.

First, the provider faces the problems grouped together in the abstraction phase, where the provider should model resources according to the kind of service(s) it will supply and the type of resources it will offer. The next two challenges are faced in the scope of the operational phase. When requests arrive, a resource allocation system should be aware of the current status of resources in order to determine if there are available resources in the fog that could satisfy the present request. With information regarding fog resource availability at hand, a set of appropriate candidates may be highlighted. Based on the above information, a resource allocation system is then able to optimize already allocated resources, and can also elect available resources to fulfil future demands. The resource selection process finds a configuration that fulfils all requirements and optimizes the usage of the infrastructure.

Resource selection strategies fall into a priori and a posteriori classes. In the a priori case, the first allocation solution is an optimal one. To achieve this goal, the approach should consider all variables influencing the allocation. For example, considering virtual machine (VM) instances being allocated, the optimization approach should figure out the problem, presenting a solution (or a set of possibilities) that satisfies all constraints and meets the goals (e.g. minimization of reallocations) in an optimal manner. In an a posteriori case, once an initial allocation that can be a suboptimal solution is made, the provider should manage its resources in a continuous way in order to improve this solution. If necessary, decisions such as to add or reallocate resources should be made in order to optimize the system

Table 1
Notations used in the proposed resource provisioning optimization approach.

Symbol	Description
D	The set of data consumers
F	The set of fog nodes
A	The set of applications
U	The set of bandwidth units
C^p_f	Upload cost at fog node $f \in F$
$C_{ff'}$	Inter-fog communication cost between fog nodes f and f'
$\omega^a_{dff'}$	Overall expected delay for application a from consumer d going through f and f'
η^a_d	Average data arrival rate for application a from user $d \in D$
ζ^a_d	The length of data stream for application a uploaded by consumer d
$RS_{f'}$	The capacity of storage resources of fog node f'
$RC_{f'}$	The capacity of computation resources of fog node f'
Q_{df}	A binary variable indicating if consumer d is in the coverage of fog node f (Q_{df}=1) or not (Q_{df}=0)
α_{df}	A binary variable indicating if fog node f is associated to consumer d (α_{df}=1) or not (α_{df}=0)
α^u_{df}	A binary variable indicating if bandwidth unit u in fog node f is allocated to consumer d (α^u_{df}=1) or not ($\alpha^u{}_{df}$=0)
$\beta^a_{ff'}$	A binary variable indicating if data from fog node f for application a is processed in fog node f' ($\beta^a_{ff'}$=1) or not ($\beta^a_{ff'}$=0)
$\beta^a_{f'}$	A binary variable indicating if a VM for application a is placed in fog node f' ($\beta^a_{f'}$=1) or not ($\beta^a_{f'}$=0)
$\gamma^a_{ff'}$	The request rate for application a from fog node f to fog node f'
$\delta^a_{f'}$	The processing rate for application a in fog node f'

utilization or comply with fog users' requirements.

Since resource utilization and provisioning are dynamic and changing all the time, it is important that any a posteriori optimization approach quickly reaches an optimal allocation level, as a result of a few configuration trials. Furthermore, it should also be able to optimize the old ones, readjusting them according to new demand. In this case, the optimization approach may also fit with the definition of a priori and dynamic classification. Hence, we try to improve utilization of limited fog resources by proposing an optimization approach.

4.1. Network model

In this section, we introduce the network model of the proposed fog architecture. For the convenience of the readers, the major notations used in this paper are listed in Table 1.

As illustrated in Fig. 5, we consider an IoT network, as an undirected graph $G = (N, E)$, where the node set N includes both the data consumers set D and the fog nodes set F (i.e. $N = D \cup F$), while the edge set E represents the links between nodes. There is a link between consumer $d \in D$ and fog node $f \in F$ if and only if they are in the communication range of each other. Let binary variable Q_{df} indicate whether fog node f is reachable from user d or not. Without loss of generality, assume that all fog nodes are reachable from each other. However, the fog nodes may be located with different distances (e.g. number of hops) with each other. For fog node $f \in F$, let $C^p{}_f$ denote the upload cost and $C_{ff'}$ be the inter-fog communication cost between fog nodes f and f'. In this paper, we assume that the transmission delay $D_{ff'}$ on $e_{ff'}$ is given in advance.

For upload communications, we assume that a bandwidth unit set U is available at each fog node. Without loss of generality, all the fog nodes are with the same number of bandwidth units and all the bandwidth units are with the same size. We assume that fog node f in the network is associated with storage and computation resources with the capacity of RS_f and RC_f, respectively. To host a virtual machine (VM) for application a at fog node f, a rental cost $C^a{}_f$ is charged per time unit.

A data consumer $d \in D$ may run different applications at the same time. We denote the application set in the whole system as A. For each application, the sensed data from involved data generators shall be periodically uploaded to the associated fog node. The uploading events happen randomly and follow a Poisson distribution (Gündüz and Özsu, 2003). Let η^a_d be the average data arrival rate of application a from consumer d. The uploaded data can be processed at any fog node f', as long as there exists a corresponding VM. Most of the fog applications are usually delay-sensitive and ask for certain QoS in terms of delay. We denote the maximum tolerable delay for application a as T^a.

4.2. Problem statement

Our objective is to minimize the overall unit cost for deploying fog architecture to guarantee the required QoS of all applications from data consumers. As mentioned, cost-diversity exists among fog nodes. It is desirable to associate all data consumers to the fog node with the smallest upload communication cost. However, the number of connections is limited by the number of bandwidth units. On the other hand, a virtual machine (VM) usually requires certain resources to ensure the QoS while the resource capacities of a fog node are limited. Also,

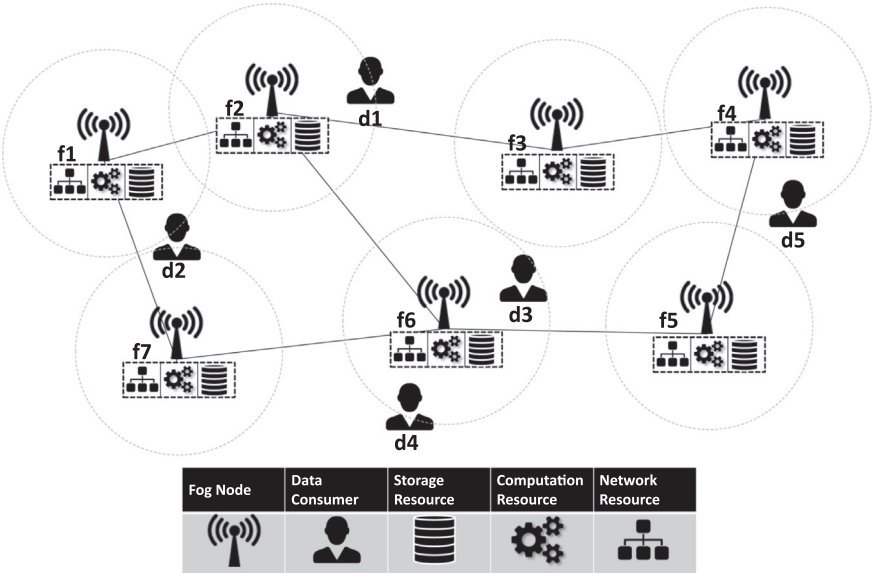

Fig. 5. An overview of the considered IoT network. Both data consumers and fog nodes are randomly located in a two-dimensional area. There is a link between consumer d and fog node f if and only if they are in the communication range of each other.

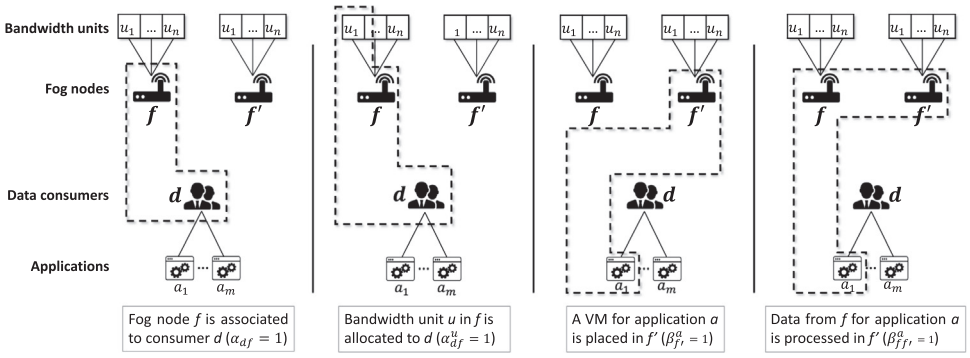

Fig. 6. Different binary variables that used in the MINLP formulation.

placing all VMs in the associated fog node help to reduce the inter-fog communication cost. But it is not efficient since the VM deployment cost of the associated fog node might be significantly higher and therefore is not ignorable. Finding the fog node association solution and an appropriate set of fog nodes to host the VMs for each application is a key issue to cost minimization. Because sensed data from data generators shall be routed from consumer-associated fog node to the fog node with the corresponding VM, it is indispensable to investigate the consumer association, task distribution and VM deployment towards cost-efficient fog architecture.

4.3. Problem formulation

In this section, we present an MINLP formulation on the minimum cost problem with the joint consideration of data consumer association, task distribution, and VM placement. Fig. 6 shows how different defined binary variables denote in the formulation.

4.3.1. Constraints of data consumer association

The binary variable α_{df}, for all $d \in D$ and $f \in F$, denotes whether data consumer d is associated with fog node f or not, i.e.,

$$\alpha_{df} = \begin{cases} 1, & \text{if data consumer } d \text{ is associated with fog node } f, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

It is worth noting that the data consumer $d \in D$ can only be associated with a fog node $f \in F$ if and only if f is reachable from data consumer d (i.e. $Q_{df} = 1$). In general, this can be expressed as

$$\sum_{u \in U} \alpha^u_{df} \le Q_{df}, \quad \forall\, d \in D,\ f \in F. \tag{2}$$

To granulate the QoS of all applications from all data consumers, one premise is that every consumer must be associated with one and only one fog node as follows:

$$\sum_{f \in F} \alpha_{df} = 1, \qquad \forall\, d \in D. \tag{3}$$

To associate a data consumer to a fog node, bandwidth units must be allocated for the communications between them. Let $\alpha^u_{df} \in \{0, 1\}$ represents allocation of bandwidth unit u at fog node f to consumer d:

$$\alpha^u_{df} = \begin{cases} 1, & \text{if bandwidth unit } u \text{ at fog node } f \text{ is allocated to consumer } d, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

A consumer d can be associated with fog node f when one or more bandwidth units at fog node f are allocated, i.e.,

$$\frac{\sum_{u \in U} \alpha^u_{df}}{|U|} \le \alpha_{df} \le \sum_{u \in U} \alpha^u_{df}, \quad \forall\, d \in D,\ f \in F,\ u \in U. \tag{5}$$

There is no restriction on the number of bandwidth units allocated to a consumer from a fog node; but one bandwidth unit can be only allocated to at most one consumer. Thus we have:

$$\sum_{d \in D} \alpha^u_{df} \le 1, \quad \forall\, f \in F,\ u \in U. \tag{6}$$

4.3.2. Constraints of task distribution

The data uploaded at the consumer-associated fog node can be processed at any fog node in the system. Let $\beta^a_{ff'}$ for all $a \in A$, $f \in F$ and $f' \in F$ represents whether data for application a uploaded through fog node f is then processed in fog nodes f', i.e.,

$$\beta^a_{ff'} = \begin{cases} 1, & \text{if data for } a \text{ from fog node } f \text{ is processed in } f', \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where f' can be equal to f if the data is processed directly at the consumer-associated fog node. Since data is uploaded through fog node f and then processed in fog node f', $\beta^a_{ff'}$ and α_{df} have the following relationship

$$\beta^a_{ff'} \le \sum_{d \in D} \alpha_{df}, \quad \forall\, a \in A, f, f' \in F, \tag{8}$$

which indicates that fog node f can distribute application data to fog node f' only when it is associated with application consumers.

Using $\gamma^a_{ff'}$ denote the data rate of application a routed from fog node f to fog node f' for processing, we obtain the relation between $\gamma^a_{ff'}$ and $\beta^a_{ff'}$ as:

$$\frac{\gamma^a_{ff'}}{N} \le \beta^a_{ff'} \le \gamma^a_{ff'} \cdot N, \quad \forall\, a \in A, f, f' \in F, \tag{9}$$

where N is an arbitrarily large number.

To ensure that all uploaded data are completely processed, for each application the total data received from data consumers through fog node f shall be equal to data finally processed at all fog node f', i.e.,

$$\sum_{d \in D} \eta^a_d \alpha_{df} = \sum_{f' \in F} \gamma^a_{ff'}, \quad \forall\, f \in F, a \in A. \tag{10}$$

4.3.3. Constraints of virtual machine (VM) Placement

Let binary variable $\beta^a_{f'}$ indicate whether a VM of application a is hosted by fog node f' or not. If data of application a is processed in f' (i.e. $\beta^a_{ff'}$=1), a corresponding VM must be deployed in f'. So, we have

$$\beta^a_{ff'} \le \beta^a_{f'}, \quad \forall\, a \in A, f, f' \in F. \tag{11}$$

A fog node can process data stream of application a if and only if a VM for a is deployed on it. Thus, we have

$$\frac{\delta^a_{f'}}{N} \le \beta^a_{f'} \le N\delta^a_{f'}, \quad \forall\, a \in A, f' \in F. \tag{12}$$

According to the resource limitation of a fog node (i.e. hard disk and computational unit), the total VM resource requirements at f' shall not exceed its capacity. Therefore, we have

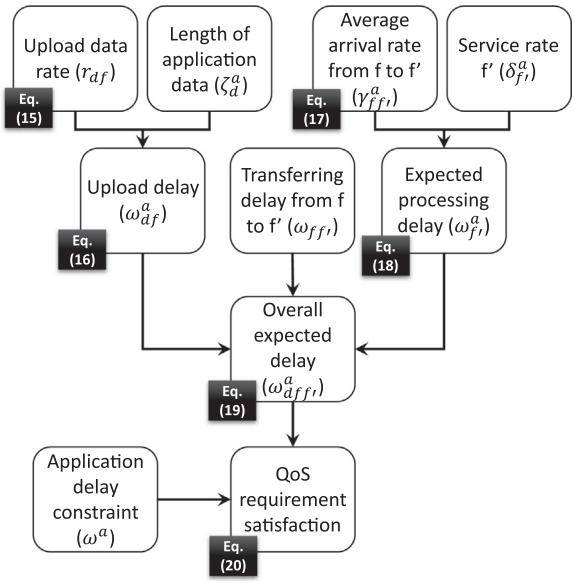

Fig. 7. Participation of the three stages of sensed data in the QoS. To satisfy the QoS requirement, the overall expected delay for any application and any consumer shall not exceed the application delay constraint.

$$\sum_{a\in A} \beta_{f'}^a RS^a \le RS_{f'}, \quad \forall f' \in F, \tag{13}$$

and

$$\sum_{a\in A} \delta_{f'}^a \nu^a \le RS_{f'}, \quad \forall f' \in F, \tag{14}$$

where $\delta_{f'}^a$ is the processing speed for application a and ν^a is a scaling factor to indicate the relation between processing speed and allocated computation resource.

4.3.4. Constraints of QoS

According to the data analysis procedure in the proposed fog architecture and as depicted in Fig. 8, the sensed data meet three stages:

1. uploading to the consumer-associated fog node f,
2. transferring from fog node f to f',
3. processing at f'.

As illustrated in Fig. 7, all of these stages influence the overall delay and hence the QoS. The upload data rate r_{df} at fog node f for consumer d is related to the number of allocated bandwidth units and thus can be calculated as

$$r_{df} = \sum_{u\in U} \alpha_{df}^u R, \quad \forall d \in D,\ f \in F, \tag{15}$$

from which we can observe that the more bandwidth units are allocated to a data consumer, the higher data rate can be achieved. We obtain upload delay for application data with length $\zeta^a{}_d$ as follows:

$$\omega_{df}^a = \frac{\zeta_d^a}{r_{df}}. \tag{16}$$

We assume that the data received by the consumer-associated fog node from consumers will be distributed to fog nodes with corresponding VMs following Uniform distribution. Thus, the data arrival in each fog node can also be regarded as a Poisson process. We define $\gamma_{df}(\gamma_{df} \le 1)$ and $\gamma_{ff'}(\gamma_{ff'} \le 1)$ as the average arrival rate of data from consumer d to fog node f and the one from fog node f to fog node f', respectively. The data processing procedure in fog node f' for application a can be viewed as a queuing system with the average arrival rate $\Sigma_{f\in F}\gamma_{ff'}$ and service rate $\delta_{f'}^a$. Based on the below condition the queuing system is stable:

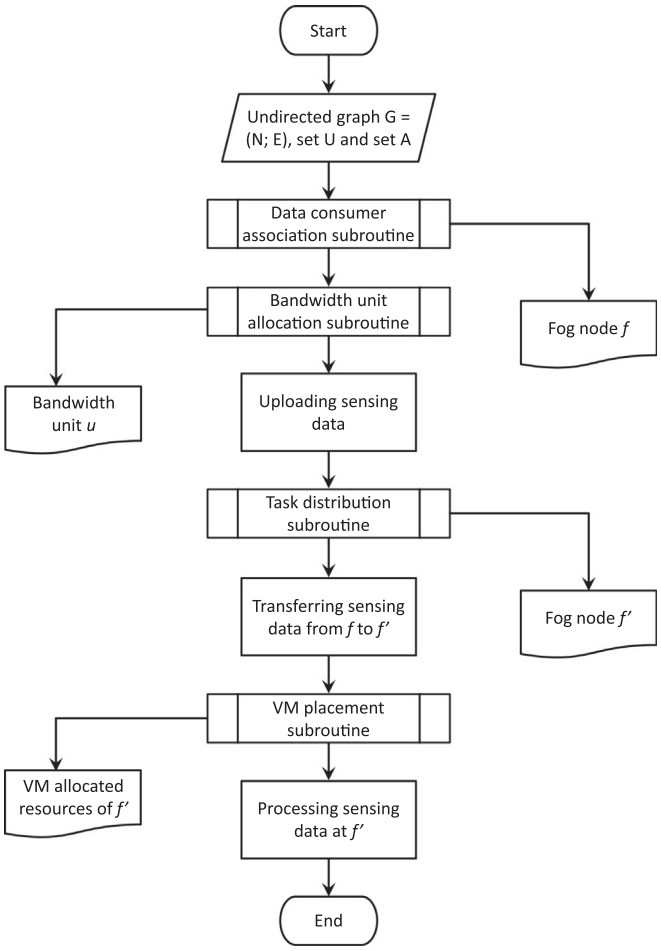

Fig. 8. The proposed flowchart for finding out the fog node association solution and an appropriate set of fog nodes to host the VMs for each application of data consumer.

$$\sum_{f\in F} \gamma_{ff'}^a < \delta_{f'}^a. \tag{17}$$

Under the existence of stability condition in the steady state, the expected processing delay $\omega_{f'}^a$ at each fog node can be written as:

$$\omega_{f'}^a = \frac{1}{\delta_{f'}^a - \sum_{f\in F} \gamma_{ff'}^a}. \tag{18}$$

Therefore, as depicted in Fig. 7, the overall expected delay for application a from consumer d going through f and f' is

$$\omega_{dff'}^a = \omega_{df}^a \alpha_{df} + \omega_{ff'} \beta_{ff'}^a + \omega_{f'}^a \beta_{f'}^a. \tag{19}$$

To satisfy the QoS requirement, the overall expected delay $\omega_{dff'}^a$ for any application and any consumer shall not exceed the application delay constraint ω^a. This leads to

$$\omega_{df}^a \alpha_{df} + \omega_{ff'} \beta_{ff'}^a + \omega_{f'}^a \beta_{f'}^a \le \omega^a,$$
$$\forall a \in A, d \in D, f, f' \in F. \tag{20}$$

By taking (16) and (18) into (20), we further have

$$\zeta_d^a\left(\alpha_{df}\delta_{f'}^a - \alpha_{df}\sum_{f\in F}\gamma_{ff'}^a\right)$$
$$+\ \omega_{ff'}R\left(\sum_{u\in U}\alpha_{df}^u\beta_{ff'}^a\delta_{f'}^a - \sum_{u\in U}\alpha_{df}^u\beta_{ff'}^a\sum_{f\in F}\gamma_{ff'}^a\right)$$
$$+\ R\sum_{u\in U}\alpha_{df}^u\beta_{ff'}^a$$
$$\le \omega^a R\left(\sum_{u\in U}\alpha_{df}^u\delta_{f'}^a - \sum_{u\in U}\alpha_{df}^u\sum_{f\in F}\gamma_{ff'}^a\right),$$
$$\forall\ a\in A,\ d\in D,\ f,f'\in F. \tag{21}$$

4.4. An MINLP formulation

When the bandwidth units are exclusively allocated to a consumer, the upload cost is determined by the upload rate r_{df}, regardless of the data volume. On the other hand, the inter-fog communication cost is determined by the actual network traffic as $\gamma_{ff'}^a\zeta_d^a$. Hence, the total communication cost can be calculated as

$$C_{com} = \sum_{d\in D}\sum_{f\in F}C_f^p r_{df} + \sum_{f\in F}\sum_{f'\in F}C_{ff'}\gamma_{ff'}^a\zeta_d^a. \tag{22}$$

Besides the communication cost, the total cost also take the VM deployment cost into consideration, i.e.,

$$C_{total} = \sum_{d\in D}\beta_{f'}^a C_f^a + C_{com},\quad \forall\ a\in A, f'\in F. \tag{23}$$

Our goal is to minimize the overall cost by choosing the best settings of α_{df}, $\alpha^u{}_{df}$, $\beta_{ff'}^a$, $\beta_{f'}^a$, $\gamma_{ff'}^a$, and $\delta_{f'}^a$. By summarizing all constraints discussed above, we can formulate this cost optimization as a mixed-integer non-linear programming (MINLP) problem.

MINLP : min (23),

s.t. (2), (3), (5), (6), (9)–(14), (17), (21),

$\alpha_{df}, \alpha_{df}^u, \beta_{ff'}^a, \beta_{f'}^a \in \{0, 1\}$,

$\forall\ a\in A, d\in D, f, f'\in F, u\in U$.

To prevent the diversion from the main focus of the paper, we will discuss about the linearisation of the MINLP problem in Appendix A.

5. Performance evaluation

5.1. Simulation setup

Modelling and simulation is the important tool to measure and evaluate the performance of any cloud model (Ahmed and Sabyasachi, 2014). In this section, we discuss about the network setup, power consumption, and the cost variables corresponding to the fog computing framework in comparison with the conventional cloud.

5.1.1. Network assumptions

As mentioned in Section 4.1, the set of essential nodes of the fog include the sets of the fog nodes F, and the data consumers D. We also define the cloud data centre set C in this section. We consider a deployment of these essential nodes in the Tehran province as the highest populated province of Iran (Brinkoff, 2016), the respective population of people using Internet services (M.M. Group, 2016), and the corresponding geographic location of the data centres (DCs), as shown in Table 2 and Fig. 9, respectively. A matrix, namely LC, stores the relative Euclidean distance between any pair of cities of Tehran province based on Table 3.

Table 2
Tehran province Internet users and population statistics 2015 (M.M. Group, 2016).

Population (Census 2011-10-24)	Population (2015 estimation)	Internet users (2015)	% Penetration
12 183 391	13 255 530	7 555 652	57.2

The data traffic generated from the cities will be proportional to the population of Internet users of the corresponding city. Data from the data generators are transmitted to the fog computing layer in the form of packets. These packet sizes typically vary between a minimum of 34 bytes (i.e. header + frame check sequence (FCS) only) to a maximum of 65 550 bytes. The instruction size is taken as 64 bits. The packet arrival is considered to follow a Poisson distribution with the mean packet arrival rate being 1 packet per node per second. Communication between the fog nodes and the cloud computing layer are assumed to take place through bandwidth unconstrained channels. However, the capacity of the links between data generators and fog nodes is considered to be 1 Gbps. On the other hand, the inter-fog communication link capacity is taken as 10 Gbps.

5.1.2. Entities assumptions

The total number of data generators in the system is treated as a variable, within the range [10 000,100 000], to assess the system performance against varying network conditions. The number of data generators in each of the cities is taken as proportional to the population of Internet users of the city. The data generators transmit their data through access points distributed within each city.

The number of data centres in Tehran province is considered to be 8. The pair-wise Euclidean distances between the data centres are stored in the matrix LD. Every data centre is assumed to accommodate the varied number of IoT devices within the discrete set {16 000, 32 000, 64 000, 128 000}, based on the network traffic is processed.

5.1.3. Power and cost assumptions

The power consumed by the different network, computing, and storage elements is used to compute the overall power consumption of the system. The power consumed by each 1 Gbps and 10 Gbps router is taken as 20 W and 40 W, respectively (Larumbe and Sansò, 2013). Each 3-layer network switch and storage component consumes 350 W and 600 W, respectively. The fog computing devices, within a fog node, consumes a total of 3.7 W collectively. Power consumption by the cloud data centres is taken to be proportional to the IoT devices within those, and taken from the range {9.7, 19.4, 38.7, 77.4} MW.

For analysis of the operational cost, we define the cost of the different component elements. Each 1 Gbps and 10 Gbps router port costs $50/year. The cost for each server is taken as $4000/year (Larumbe and Sansò, 2013). The uploading cost for each GB of data is taken as $0.12, and the storage cost per GB of data is taken uniformly from within the range [0.45–0.55]$/hour. The electricity cost is taken as uniformly distributed between $30/MW h and $70/MW h (Larumbe and Sansò, 2013).

5.2. Results analysis

In details of the results obtained in terms of the performance metrics are presented and analysed in this section. We compare the performance of the fog computing paradigm with that of the traditional cloud computing architecture, and present a thorough study against the same. Note that we adopted 95% confidence level to make sure that, on average, the confidence interval and standard error contain the true values approximately 95% of the time.

5.2.1. Service latency

First, we analyse the variation of the service latency (transmission latency + processing latency) with the number of data consumers. We define the ratio of the total bytes transmitted to the fog computing layer to the number of bytes referred to the cloud computing layer as the cloud transmission ratio, mathematically represented by the variable Δ.

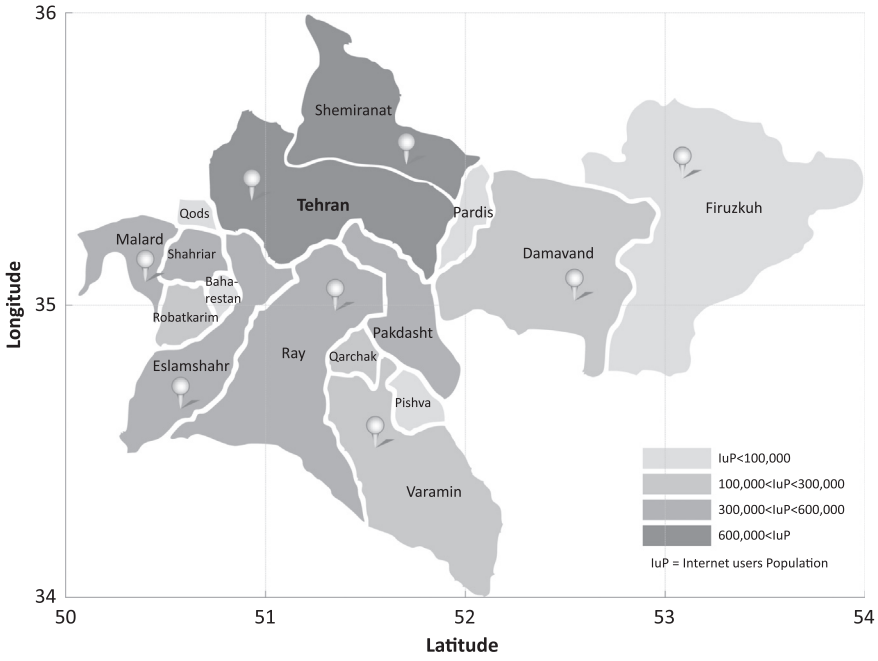

Fig. 9. Distribution of data centres and population of Internet users in the Tehran province.

Table 3
Cities of Tehran province and their corresponding geographic location.

No.	Name	Latitude	Longitude
1	Tehran	35°44′N	51°30′E
2	Shemiranat	35°78′N	51°43′E
3	Ray	35°35′N	51°25′E
4	Eslamshahr	35°40′N	51°10′E
5	Varamin	35°32′N	51°65′E
6	Shahriar	35°70′N	51°41′E
7	Damavand	35°47′N	52°00′E
8	Firuzkuh	35°50′N	52°50′E
9	Malard	35°39′N	50°58′E
10	Robatkarim	35°28′N	
11	Pakdasht	35°47′N	51°67′E
12	Pishva	35°31′N	51°73′E
13	Pardis	35°44′N	51°48′E
14	Qods	35°71′N	51°12′E
15	Qarchak	35°48′N	51°53′E
16	Baharestan	35°33′N	51°09′E

Table 4
The changes of the magnitude of Δ.

Name	Δ value	Type
Rate1	$\Delta = 0.25$	Fog
Rate2	$\Delta = 0.5$	Fog
Rate3	$\Delta = 0.75$	Fog
Rate4	$\Delta = 1$	Cloud

We consider the changes of the magnitude of Δ in the range [0.25, 1] as shown in Table 4, and plot the transmission latency and processing latency, and observe the change in the corresponding service latencies.

In Fig. 10(a) and (b), the mean transmission latency and mean processing latency are plotted, separately, against a variable number of data consumers, for different magnitudes of Δ. It is observed that with the decrease in the magnitude of Δ, as more number of application requests demand real-time and latency-sensitive services, the mean transmission latency and the mean processing latency are observed to diminish. Also, with the increase in the number of data consumers, the transmission and processing latencies increase.

Consequently, the overall service latency for both these computing paradigms follow the same pattern. In summary, for a very low percentage of applications which demand real-time services (i.e. for a very high magnitude of $\Delta(\Delta \simeq 1)$), the service latency in fog computing becomes almost the same with that in a cloud computing environment.

5.2.2. Power consumption

The power consumption due to the effects of data forwarding, computation, and data storage, and due to their collective effect is analysed in this subsection. As shown in Fig. 11(a), although with the increase in the number of data consumers, the mean power consumption due to data forwarding increases linearly, the impact of the change in the power consumption for different magnitudes of Δ is observed to be very low. However, compared to a conventional cloud computing framework (i.e. $\Delta = 1$), this mean power consumption was significantly less.

In Fig. 11(b) and (c), the variation in the mean power consumption due to computation and storage, respectively, are shown against the change in the number of data consumers. Similar inferences are drawn from the two figures, as the power consumption in both these cases are noticed to decrease by a significant margin as the magnitude of Δ is decreased.

5.2.3. Cost

As the third parameter, we analyse the cost incurred in both cloud and fog computing environments for various operations. Fig. 12 indicates the standard cost incurred. Based on the power consumption due to computation, we evaluate the cost of computation. Similarly, we also evaluate the cost for routing and storage. Both the costs are observed to be considerably higher in cloud environments than fog environments.

6. Conclusion and future work

In this paper, we presented MIST, a fog-based data analytics scheme with cost-efficient resource provisioning optimization approach that can be used for IoT crowd sensing purposes. MIST scheme is able to support context-aware quick response at three different levels (i.e. edge, fog, and cloud level), providing high computing performance, and

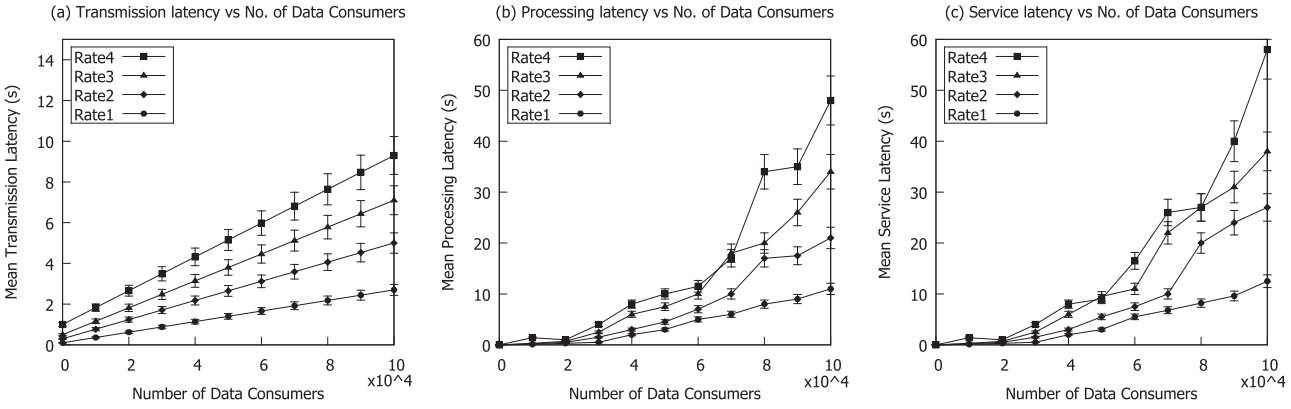

Fig. 10. Analysis of service latency.

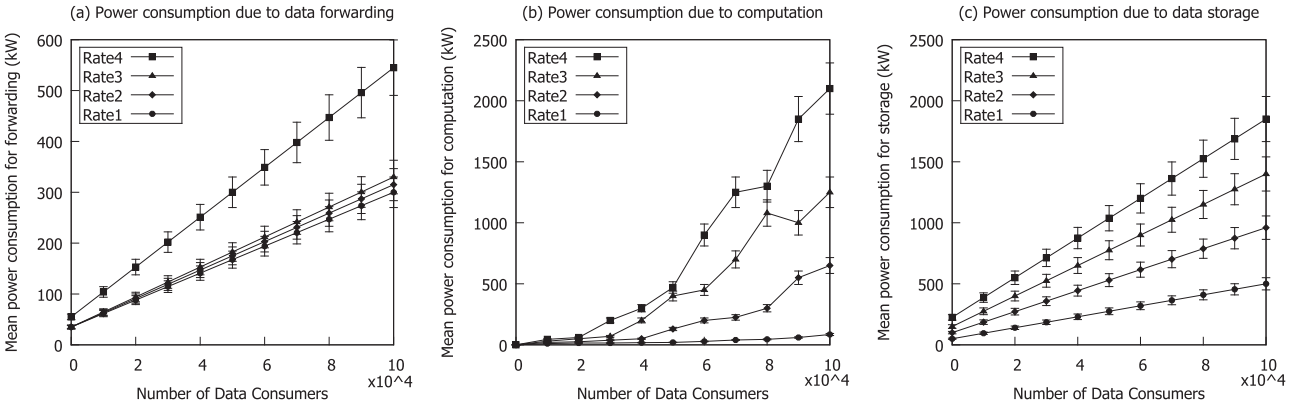

Fig. 11. Analysis of power consumption.

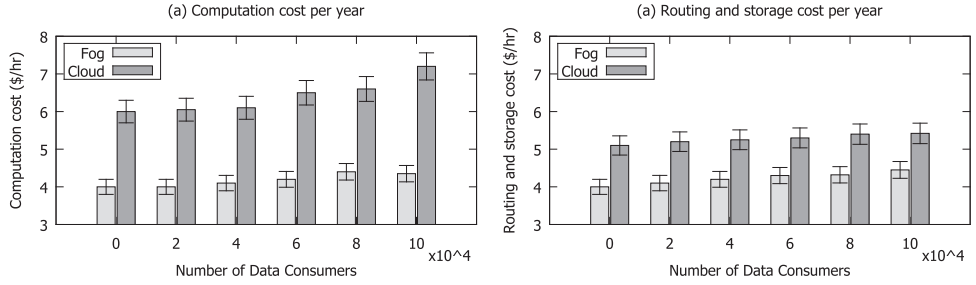

Fig. 12. Operational cost per year.

intelligence in future smart cities. Further, we focused on optimum and cost-efficient resource provisioning in the MIST scheme. To tackle the cost-efficiency problem in MIST, we jointly study data consumer association, task distribution, and virtual machine (VM) placement issues towards minimizing the overall cost while satisfying the QoS requirement. Specially, the problem is first formulated into an MINLP problem. To deal with the high computational complexity of solving MINLP, we linearise it into an MILP problem.

Finally, we perform a comparative performance evaluation of cloud computing with that of fog computing for a city with a high number of Internet-connected devices demanding different services (i.e. Tehran province). The results clearly depict the enhanced performance of fog computing in terms of the provisioned QoS. This observed performance of the fog computing architecture indicates its substantial potential as a method of future smart city monitoring and control. Some of the additional research issues that can be investigated for future extension of the work are as follows: (1) adding selective sensing module to the fog layer, (2) enriching the architecture with privacy-preserving data analytics capabilities, (3) considering the mobility of data generators and data consumers in the resource provisioning and its influence on the performance of proposed scheme, (4) conducting real-world trials.

Appendix A. Linearisation of the MINLP problem

It is difficult to solve the MINLP problem directly due to its high computational complexity. Fortunately, we notice that (21) is a cubic inequation. The two cubic terms are products of two binary variables and one floating point variable. It is possible to linearise them to lower the computational complexity. To this end, we first define a new auxiliary variable Π as follows:

$$\Pi = \alpha^u_{df}\beta^a_{ff'}, \quad \forall\, a \in A, f, f' \in F, u \in U, \tag{24}$$

which can be equivalently replaced by the following linear constraints:

$$\alpha^u_{df} + \beta^a_{ff'} - 1 \le \Pi \le \alpha^u_{df}, \quad \forall\, a \in A, f, f' \in F, u \in U, \tag{25}$$

$$\Pi \le \beta^a_{ff'}, \quad \forall\, a \in A, f \in F, f' \in F, u \in U. \tag{26}$$

Constraint (21) then becomes a quadratic in-equation with product terms of one binary variable and one floating-point variable as follows:

$$\zeta^a_d\left(\alpha_{df}\delta^a_{f'} - \alpha_{df}\sum_{f\in F}\gamma^a_{ff'}\right) + \omega_{ff'}R\sum_{u\in U}\Pi\left(\delta^a_{f'} - \sum_{f\in F}\gamma^a_{ff'}\right) + R\sum_{u\in U}\Pi \le \omega^a R\left(\sum_{u\in U}\alpha^u_{df}\delta^a_{f'} - \sum_{u\in U}\alpha^u_{df}\sum_{f\in F}\gamma^a_{ff'}\right),$$
$$\forall\, a \in A, d \in D, f, f' \in F. \tag{27}$$

Let $\Gamma = \alpha_{df}\delta^a_{f'}$, which can be equivalently rewritten into the following linear constraints:

$$0 \le \Gamma \le \delta^a_{f'}, \quad \forall\, a \in A, d \in D, f, f' \in F, \tag{28}$$

$$\delta^a_{f'} + \alpha_{df} - 1 \le \Gamma \le \alpha_{df}, \quad \forall\, a \in A, d \in D, f, f' \in F. \tag{29}$$

Similarly, letting $\Phi = \alpha_{df}\gamma^a_{ff'}$, $\Upsilon = \Pi\delta^a_{f'}$, $\Omega = \Pi\gamma^a_{ff'}$, $\Theta = \alpha^u_{df}\delta^a_{f'}$ and $\Psi = \alpha^u_{df}\gamma^a_{ff'}$, we have the following new linear constraints:

$$0 \le \Phi \le \gamma^a_{ff'}, \quad \forall\, a \in A, d \in D, f, f' \in F, \tag{30}$$

$$\alpha_{df} + \gamma^a_{ff'} - 1 \le \Phi \le \alpha_{df}, \quad \forall\, a \in A, d \in D, f, f' \in F. \tag{31}$$

$$0 \le \Upsilon \le \delta^a_{f'}, \quad \forall\, a \in A, d \in D, f, f' \in F, u \in U, \tag{32}$$

$$\Pi + \delta^a_{f'} - 1 \le \Upsilon \le \Pi, \quad \forall\, a \in A, d \in D, f, f' \in F, \ u \in U. \tag{33}$$

$$0 \le \Omega \le \gamma^a_{ff'}, \quad \forall\, a \in A, d \in D, f, f' \in F, u \in U, \tag{34}$$

$$\Pi + \gamma^a_{ff'} - 1 \le \Omega \le \Pi, \quad \forall\, a \in A, d \in D, f, f' \in F\ u \in U. \tag{35}$$

$$0 \le \Theta \le \delta^a_{f'}, \quad \forall\, a \in A, d \in D, f, f' \in F, u \in U, \tag{36}$$

$$\alpha^u_{df} + \delta^a_{f'} - 1 \le \Theta \le \alpha^u_{df}, \quad \forall\, a \in A, d \in D, f, f' \in F\ u \in U. \tag{37}$$

$$0 \le \Psi \le \gamma^a_{ff'}, \quad \forall\, a \in A, d \in D, f, f' \in F, u \in U, \tag{38}$$

$$\alpha^u_{df} + \gamma^a_{ff'} - 1 \le \Psi \le \alpha^u_{df}, \quad \forall\, a \in A, d \in D, f, f' \in F\ u \in U. \tag{39}$$

Hence, (21) can be written in a linear form as:

$$\zeta^a_d\left(\Gamma - \sum_{f\in F}\Phi\right) + \omega_{ff'}R\left(\sum_{u\in U}\Upsilon - \sum_{u\in U}\sum_{f\in F}\Omega\right) + R\sum_{u\in U}\alpha^a_{dff'} \le \omega^a R\left(\sum_{u\in U}\Theta - \sum_{u\in U}\sum_{f\in F}\Psi\right), \quad \forall\, a \in A, d \in D, f, f' \in F. \tag{40}$$

Finally, our MINLP problem can be linearised into a mixed-integer linear programming (MILP) below.

MILP : min (23), s.t. (2), (3), (5), (6), (9)–(14), (17), (25)–(29), (30)–(39), (40), $\alpha_{df}, \alpha^u_{df}, \beta^a_{ff'}, \beta^a_{f'} \in \{0, 1\}$, $\forall\, a \in A, d \in D, f, f' \in F, u \in U$.

References

Aazam, M., Huh, E.N., 2015. Dynamic resource provisioning through fog micro datacenter. In: 2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), pp. 105–110.

Abdallah, Z.S., Gaber, M.M., Srinivasan, B., Krishnaswamy, S., 2012. Streamar: incremental and active learning with evolving sensory data for activity recognition. In: 2012 IEEE 24th International Conference on Tools with Artificial Intelligence, vol. 1, pp. 1163–1170.

Agarwal, S., Dunagan, J., Jain, N., Saroiu, S., Wolman, A., Bhogan, H., 2010. Volley: automated data placement for geo-distributed cloud services. In: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI'10. USENIX Association, San Jose, California, pp. 2–8.

Agarwal, S., Yadav, S., Yadav, A.K., 2016. An efficient architecture and algorithm for resource provisioning in fog computing. Int. J. Inf. Eng. Electron. Bus. 8 (1), 48–61.

Ahmed, A., Sabyasachi, A.S., 2014. Cloud computing simulators: a detailed survey and future direction. In: 2014 IEEE International Advance Computing Conference (IACC), pp. 866–872.

Ahmad, M., Amin, M.B., Hussain, S., Kang, B.H., Cheong, T., Lee, S., 2016. Health fog: a novel framework for health and wellness applications. J. Supercomput., 1–19.

Arkian, H.R., Atani, R.E., Pourkhalili, A., Kamali, S., 2014. Cluster-based traffic information generalization in vehicular ad-hoc networks. Veh. Commun. 1 (4), 197–207.

Arkian, H.R., Atani, R.E., Diyanat, A., Pourkhalili, A., 2015. A cluster-based vehicular cloud architecture with learning-based resource management. J. Supercomput. 71 (4), 1401–1426.

C. Bits. Street Bump 2013. URL ⟨http://streetbump.org/⟩

Bonomi, F., Milito, R., Zhu, J., Addepalli, S., 2014a. Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on

Mobile Cloud Computing, MCC '12. ACM, New York, NY, USA, 2012, pp. 13–16.
Bonomi, F., Milito, R., Natarajan, P., Zhu, J., 2014b. Fog computing: a platform for internet of things and analytics. In: Big Data and Internet of Things: A Roadmap for Smart Environments. Springer International Publishing, Cham, pp. 169–186.
Botta, A., de Donato, W., Persico, V., Pescape, A., 2016. Integration of cloud computing and internet of things: a survey. Future Gener. Comput. Syst. 56 (1), 684–700.
Brinkoff, T., 2016. City population. URL ⟨http://www.citypopulation.de/⟩
Cavalcante, E., Pereira, J., Alves, M.P., Maia, P., 2016. On the interplay of internet of things and cloud computing: a systematic mapping study. J. Comput. Commun. 89 (9), 17–33.
Chiang, M., Zhang, T., 2016. Fog and IoT: an overview of research opportunities. IEEE Internet of Things J. 3 (6), 854–864.
P.G. Corporation, 2013. SmartThings Hub User Guide. URL ⟨https://www.smartthings.com/⟩
Diaz, M., Martin, C., Rubio, B., 2016. State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing. J. Netw. Comput. Appl. 67, 99–117.
Do, C.T., Tran, N.H., Pham, C., Alam, M.G.R., Son, J.H., Hong, C.S., 2015. A proximal algorithm for joint resource allocation and minimizing carbon footprint in geo-distributed fog computing. In: 2015 International Conference on Information Networking (ICOIN), pp. 324–329.
Endo, P.T., de Almeida Palhares, A.V., Pereira, N.N., Goncalves, G.E., Sadok, D., Kelner, J., Melander, B., Mangs, J.E., 2011. Resource allocation for distributed cloud: concepts and research challenges. IEEE Netw. 25 (4), 42–46.
Ganti, R.K., Ye, F., Lei, H., 2011. Mobile crowdsensing: current state and future challenges. IEEE Commun. Mag. 49 (11), 32–39.
M.M. Group, 2016. Internet world stats. URL ⟨http://www.internetworldstats.com/⟩
Gu, L., Zeng, D., Guo, S., Barnawi, A., Xiang, Y., 2015. Cost-efficient resource management in fog computing supported medical cps. IEEE Trans. Emerg. Top. Comput. PP (99), 1–12.
Gündüz, Ş., Özsu, M.T., 2003. A Poisson model for user accesses to web pages. In: Computer and Information Sciences – ISCIS 2003: 18th International Symposium, Antalya, Turkey, November 3–5, 2003. Proceedings. Springer, Berlin, Heidelberg, pp. 332–339.
Hasenfratz, D., Saukh, O., Sturzenegger, S., Thiele, L., 2012. Participatory air pollution monitoring using smartphones. In: Mobile Sensing: From Smartphones and Wearables to Big Data. ACM, Beijing, China.
Hong, K., Lillethun, D., Ramachandran, U., Ottenwälder, B., Koldehofe, B., 2013. Mobile fog: a programming model for large-scale applications on the internet of things. In: Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing, MCC '13. ACM, New York, NY, USA, pp. 15–20.
Jayaraman, P.P., Gomes, J.B., Nguyen, H.L., Abdallah, Z.S., Krishnaswamy, S., Zaslavsky, A., 2015. Scalable energy-efficient distributed data analytics for crowdsensing applications in mobile environments. IEEE Trans. Comput. Soc. Syst. 2 (3), 109–123.
Larumbe, F., Sansò, B., 2013. A tabu search algorithm for the location of data centers and software components in green cloud computing networks. IEEE Trans. Cloud Comput. 1 (1), 22–35.
Perera, C., Talagala, D.S., Liu, C.H., Estrella, J.C., 2015a. Energy-efficient location and activity-aware on-demand mobile distributed sensing platform for sensing as a service in iot clouds. IEEE Trans. Comput. Soc. Syst. 2 (4), 171–181.
Perera, C., Liu, C.H., Jayawardena, S., 2015b. The emerging Internet of things marketplace from an industrial perspective: a survey. IEEE Trans. Emerg. Top. Comput. 3 (4), 585–598.
Preden, J., Kaugerand, J., Suurjaak, E., Astapov, S., Motus, L., Pahtma, R., 2015. Data to decision: pushing situational information needs to the edge of the network. In: 2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision, pp. 158–164.
Qin, Y., Sheng, Q.Z., Falkner, N.J., Dustdar, S., Wang, H., 2016. When things matter: a survey on data-centric internet of things. J. Netw. Comput. Appl. 64 (4), 137–153.
Sarkar, S., Chatterjee, S., Misra, S., 2015. Assessment of the suitability of fog computing in the context of internet of things. IEEE Trans. Cloud Comput. PP (99), 1–14.
Satyanarayanan, M., Schuster, R., Ebling, M., Fettweis, G., Flinck, H., Joshi, K., Sabnani, K., 2015. An open ecosystem for mobile-cloud convergence. IEEE Commun. Mag. 53 (3), 63–70.
Schappi, M., 2013. How to setup the Ninja Block software image using linux. URL ⟨https://ninjablocks.com/⟩
Scuotto, V., Ferraris, A., Bresciani, S., 2016. Internet of things: applications and challenges in smart cities: a case study of ibm smart city projects. Bus. Process Manag. J. 22 (2), 357–367.
Sherchan, W., Jayaraman, P.P., Krishnaswamy, S., Zaslavsky, A., Loke, S., Sinha, A., 2012. Using on-the-move mining for mobile crowdsensing. In: 2012 IEEE 13th International Conference on Mobile Data Management, pp. 115–124.
I. Streetline, 2014. ParkSight: The complete Smart Parking solution. URL ⟨https://www.streetline.com/parking-analytics/⟩
Tang, B., Chen, Z., Hefferman, G., Wei, T., He, H., Yang, Q., 2015. A hierarchical distributed fog computing architecture for big data analysis in smart cities. In: Proceedings of the ASE BigData & Social Informatics 2015, ASE BD&SI '15. ACM, New York, NY, USA, pp. 28:1–28:6.
Urgaonkar, R., Kozat, U.C., Igarashi, K., Neely, M.J., 2010. Dynamic resource allocation and power management in virtualized data centers. In: 2010 IEEE Network Operations and Management Symposium – NOMS 2010, pp. 479–486.
Willis, D.F., Dasgupta, A., Banerjee, S., 2014. Paradrop: a multi-tenant platform for dynamically installed third party services on home gateways. In: Proceedings of the 2014 ACM SIGCOMM Workshop on Distributed Cloud Computing, DCC '14. ACM, New York, NY, USA, pp. 43–44.
Yi, S., Hao, Z., Qin, Z., Li, Q., 2015. Fog computing: platform and applications. In: 2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb), 2015, pp. 73–78.
Yi, S., Qin, Z., Li, Q., 2015. Security and privacy issues of fog computing: a survey. In: Wireless Algorithms, Systems, and Applications: 10th International Conference, WASA 2015, Qufu, China, August 10–12, 2015, Proceedings. Springer International Publishing, Cham, pp. 685–695.
Zhu, J., Chan, D.S., Prabhu, M.S., Natarajan, P., Hu, H., Bonomi, F., 2013. Improving web sites performance using edge servers in fog computing architecture. In: 2013 IEEE 7th International Symposium on Service Oriented System Engineering (SOSE), pp. 320–323.

Hamid Reza Arkian received the M.Sc. degree in Information Technology Engineering from the University of Guilan, Rasht, Iran, in 2014. He is currently a research assistant in Computer Engineering Department of University of Guilan. He has published over 15 papers in peer reviewed national and international journal and conferences. His current research interests include wireless networks, vehicular ad hoc networks (VANETs), distributed computing, and Internet of things.

Abolfazl Diyanat received his B.Sc. degree in Electrical and Computer Engineering from the University of Tehran, Tehran, Iran, in 2009, and M.Sc. degree in Communication Engineering from Sharif University of Technology, Tehran, Iran. He is now a Ph.D. candidate in the Department of Electrical and Computer Engineering at the University of Tehran. His main research interests include telecommunication systems, statistical privacy in the network, security, and analytical modelling of wireless ad hoc networks.

Atefe Pourkhalili received her B.Sc. and M.Sc. degrees in Information Technology Engineering from the University of Guilan, Rasht, Iran, in 2012 and 2014, respectively. She is currently a research assistant in Computer Engineering Department of University of Guilan. Her main research interests include ad hoc networks, wireless sensor networks, and Internet of things.