IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 30, 2017, accepted August 7, 2017, date of publication August 15, 2017, date of current version September 6, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2739804

INVITED PAPER

Edge Mesh: A New Paradigm to Enable Distributed Intelligence in Internet of Things

YUVRAJ SAHNI[1], JIANNONG CAO[1], (Fellow, IEEE), SHIGENG ZHANG[2], (Member, IEEE), AND LEI YANG[3]

[1]Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[2]School of Information Science and Engineering, Central South University, Changsha 410000, China
[3]School of Software Engineering, South China University of Technology, Guangzhou 510640, China

Corresponding author: Yuvraj Sahni (csysahni@comp.polyu.edu.hk)

This work was supported in part by the RGC General Research Fund under Grant PolyU 152244/15E, in part by the NSFC Key Project Grant 61332004, and in part by the National Natural Science Foundation of China under Grant 61502312.

ABSTRACT In recent years, there has been a paradigm shift in Internet of Things (IoT) from centralized cloud computing to edge computing (or fog computing). Developments in ICT have resulted in the significant increment of communication and computation capabilities of embedded devices and this will continue to increase in coming years. However, existing paradigms do not utilize low-level devices for any decision-making process. In fact, gateway devices are also utilized mostly for communication interoperability and some low-level processing. In this paper, we have proposed a new computing paradigm, named Edge Mesh, which distributes the decision-making tasks among edge devices within the network instead of sending all the data to a centralized server. All the computation tasks and data are shared using a mesh network of edge devices and routers. Edge Mesh provides many benefits, including distributed processing, low latency, fault tolerance, better scalability, better security, and privacy. These benefits are useful for critical applications, which require higher reliability, real-time processing, mobility support, and context awareness. We first give an overview of existing computing paradigms to establish the motivation behind Edge Mesh. Then, we describe in detail about the Edge Mesh computing paradigm, including the proposed software framework, research challenges, and benefits of Edge Mesh. We have also described the task management framework and done a preliminary study on task allocation problem in Edge Mesh. Different application scenarios, including smart home, intelligent transportation system, and healthcare, are presented to illustrate the significance of Edge Mesh computing paradigm.

INDEX TERMS Edge devices, Internet of Things, distributed intelligence, distributed computing, mesh network.

I. INTRODUCTION

Internet of Things (IoT) envisions to revolutionize our life by connecting everything around us with each other. IoT has changed the way we think about our surrounding. IoT affects almost all aspects of our life including our homes, offices, healthcare, transportation, power grid, logistics, industries, and many more areas. Most of the IoT applications can be abstracted as shown in Fig. 1, where there are four main components i.e. Sensing, Communication, Computation, and Actuation. IoT envisions embedding of sensing/communication/computation/actuation capabilities in common objects, however, in existing systems, a single device usually does not supports all the capabilities. Most IoT systems use end devices for sensing the surroundings while communication and networking responsibilities are undertaken by gateways and routers. Computation is usually done at a centralized server and the information generated by processing is utilized by some selected devices that act as actuators. Sensing, communication and networking have always been the focus of attention for researchers, however, researchers have now also started considering issues related to computation and intelligence. As the number of devices continues to increase in the coming future, a major issue will be to generate useful information through computation.

Computation is an important part of IoT as it leads to knowledge generation, which can be utilized to provide better and intelligent services. Terms such as ''Smart'' or ''Intelligent'' are usually associated with IoT, however, it is not

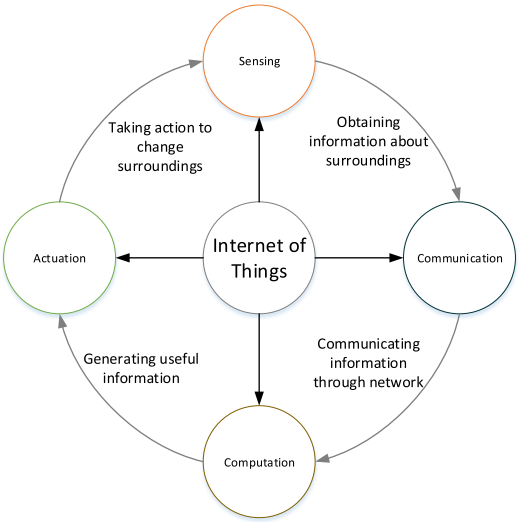


FIGURE 1. Overview of generic IoT system.

clearly defined what is intelligence, in the context of IoT? and who provides it? These types of questions have motivated researchers to study about topics such as data-centric IoT [1], data mining in IoT [2], the interaction of artificial intelligence with IoT etc. Many terms such as Context-awareness [3], autonomous control, ambient intelligence, cognitive IoT [4], semantic reasoning [5] etc. are associated with intelligence in IoT but each term only covers part of the whole picture. Due to rapid development in technology, scope and definition of intelligence gets widened and evolved. An example of this is smart home products which have evolved to an extent that they can understand context, surroundings, and even respond accordingly.

Most of the existing IoT systems use a centralized server for computation purpose. Low-level end devices are utilized only for sensing purpose and the decision-making is done by a centralized server which collects data from multiple devices. Even gateway devices which have more processing and storage power are used mostly for data aggregation and low-level processing. This centralized computing paradigm not only leads to wastage of resources but also is not efficient for computation-intensive and time-critical applications. In this paper, we propose Edge Mesh which can be defined as a computing paradigm that uses mesh network of Edge devices and routers to enable distributed decision-making within the network. The decision-making is done inside the network by sharing data and computation among Edge devices instead of sending all the data to a centralized server. This is different from existing computing paradigm which usually perform centralized computing and Edge devices such as gateways are used only for collecting and transmitting the data to a server for processing.

Edge Mesh proposes the idea of using Edge devices to enable distributed intelligence in IoT. In [6], distributed intelligence is defined as ''cooperation between devices, intermediate communication infrastructures (local networks, access networks, global networks) and/or cloud systems in order to optimally support IoT communication and IoT applications''. They consider that distributed intelligence involves both processing and networking elements. We consider distributed intelligence in a more broader perspective which involves everything from data analytics and networking to other functionalities such as data management, device management, resource management, service management, orchestration, etc. There are many research questions to be answered to develop Edge Mesh such as: How to define the network and computing model? How to distribute processing of data? How to jointly optimize communication and computation? etc. Existing computation algorithms cannot be directly used in case of Edge Mesh as distributed computation is done by Edge devices which are heterogeneous, resource-constraint, and have a dynamic communication channel with intermittent connectivity [7]. Other issues related to IoT such as heterogeneity, reliability, fault tolerance, QoS management, security and privacy etc. [8] must also be considered.

Most of the existing works in IoT focus on centralized computation, but there are some works such as [6], [9], and [10] that also consider distributed computation. Van den Abeele *et al.* [6] propose the idea of using sensor function virtualization to support distributed intelligence in IoT. Our work considers distributed intelligence from a broader perspective and our proposed idea to enable distributed intelligence is also different. A vision of distributed actuation and in-network processing has also been discussed in [9], however, it does not provide any details. Another closely related work is Symbiot, which proposes an architecture where all devices are logically connected in a mesh network [10]. The work proposed in [10] does not focus on distributed intelligence and only the high-level details about the proposed architecture are presented. Many challenges including task management, data management etc. have not been considered in Symbiot [10].

The main contributions of this paper are:

- A new computing paradigm, Edge Mesh, which focuses on enabling distributed Intelligence in IoT has been proposed.
- Motivation, software framework, challenging research issues, and benefits of Edge Mesh have been discussed.
- Task Allocation problem in Edge Mesh has been studied. The task allocation problem considers distribution of data for dependent tasks that are allocated to set of devices such that total energy consumption is minimized. To the best of our knowledge, we are the first ones to consider data distribution, task dependency, embedded device constraint, and device heterogeneity simultaneously for task allocation problem.
- Application scenarios have been given to illustrate the significance of Edge Mesh for different applications domains.

The rest of the paper is as follows. Section 2 discusses existing computing paradigms in IoT and motivation for proposing Edge Mesh. Section 3 discusses in detail about

the hardware and software architecture, and benefits of Edge Mesh. Section 4 discusses the task management framework as well as task allocation problem in Edge Mesh. Section 5 discusses applications scenarios in different application domains such as Smart Home, HealthCare, and Intelligent Transportation System that can benefit from Edge Mesh. Section 6 discusses the open issues and challenges associated with Edge Mesh. In Section 7, we conclude the paper with conclusion and future work.

II. BACKGROUND AND MOTIVATION

IoT is predicted to have 50 billion devices by 2020. Computing paradigms for IoT must handle the huge scale of devices and application areas. This variability in application and system requirements leads to different views about computation paradigms for IoT. Besides, computing in IoT is not an independent area, it impacts other aspects too such as communication, energy efficiency, physical design, software development, analytics, user experience, security etc. [11]. All these aspects together contribute towards intelligence in IoT. In this section, we give a brief account of three main computing paradigms that are currently being used in IoT, i.e. centralized cloud computing, fog computing, and cooperative computing. We also discuss some issues in existing computing paradigms that have motivated us to propose Edge Mesh.

A. CENTRALIZED CLOUD COMPUTING

NIST defines Cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [12]. IoT devices are resource constraint which limits their applicability, on the other hand, Cloud has abundant resources. Therefore, IoT can make use of resources in the cloud to make up for its limited resources [13]. Cloud can benefit IoT in many ways including communication, computation, and storage. Data collected by IoT devices can be stored in cloud and processed in a low-cost and effective manner. Cloud is especially useful for IoT applications that are computation-intensive (i.e. use complex analytics algorithms which cannot run on resource-constraint devices) and/or use data-driven processing [13]. Ambient intelligence application that requires machine learning algorithm belongs to such category of IoT applications.

Cloud computing paradigm is heavily dependent on Internet connectivity. Due to intermittent network connectivity, network latency becomes high which is not suitable for applications with the real-time requirement. As the amount of data being generated by IoT devices is becoming huge, it is very difficult to send all the data to Cloud due to limited bandwidth constraint. There is also a major issue of security and privacy as data travels along intermediate networks which can be prone to attacks and if the data is stored at public cloud then chances of unwanted access and/or compromise becomes higher. Since Cloud data centers are usually located at a faraway place, latency is higher which means Cloud computing paradigm is not effective for an application that requires mobility support.

B. FOG COMPUTING

Cloud computing paradigm suffers from four major issues discussed above, i.e. latency, security, privacy, and mobility, which has motivated researchers to propose a Fog Computing paradigm. Open Fog Consortium defines Fog Computing as "a system level horizontal architecture that distributes resources and services of computing, storage, control and networking anywhere along the continuum from cloud to Thing" [14]. Fog Computing is a distributed paradigm that provides characteristics such as low latency, real-time interaction, distributed analytics, context awareness, geographical distribution, mobility support, which are not supported by centralized Cloud computing paradigm [15]. Fog Computing shares many similarities with Edge Computing paradigm as both allow computation closer to devices that produce data [16], [17]. Fog computing focuses on infrastructure perspective while Edge computing focuses on things perspective [17]. There is another similar paradigm called Mobile Edge Computing, which was proposed by ETSI as a platform that pushes cloud computing capabilities closer to mobile devices in radio access networks (RAN) [18].

There are many challenging issues that need to be resolved to realize the full potential of fog computing paradigm. These issues are related to fog networking, Quality of service, interfacing and programming, computation offloading, accounting, billing, monitoring, provisioning and resource management, and security and privacy [19]. IoT integrates different application domains which can have varying requirements. Fog computing and cloud computing paradigms both provide different benefits but they can work complementary with each other to satisfy multiple applications requirements [15]. Currently researchers are working to integrate Fog and Cloud computing paradigms [20], [21]. Osmotic computing paradigm proposed in [20] aims to decompose applications into microservices and use resources in both edge and cloud to dynamically satisfy application requirements. IFCIoT is another work that has been proposed related to integration where federated cloud services are provided by intermediary fog layer [21]. Distributed fog nodes collect data from local systems and updated data is then sent to federated cloud data center which can further perform big data analytics to give a globalized view of whole system [21]. Fog computing can also be incorporated with emerging networking technologies such as 5G technologies, network function virtualization (NFV), and software-defined networking (SDN) [22].

C. COOPERATIVE COMPUTING

Fog computing, edge computing and other similar paradigms have been proposed to deal with issues of latency, mobility, bandwidth bottleneck etc. in traditional cloud computing.

Even in fog computing, the data from local devices is sent to a local server located near to devices. In coming future, the number of devices will be very high which will lead to heavy load on servers and fog devices. Mobile devices available in the market have high computation, storage and communication capabilities which have led to an emerging area of opportunistic cooperation among mobile devices. Instead of offloading the tasks to edge computing servers, mobile devices can make use of surrounding devices for task processing. This will solve a major issue of overload on edge computing servers. Many times, few mobile devices are overloaded with computation tasks while other devices are free which not only leads to uneven distribution of tasks but also increased energy consumption and latency. Cooperative computing can also help in resolving this issue of unbalanced computation distribution [18].

Cooperative computing provides many benefits to IoT systems including better usage of resources, reduced latency due to easy access to local resources, better services as devices can cooperate with each other to get better information, reduced communication with cloud and other outside entities, and improved security and privacy as data will usually remain within a local network. Applications such as smart home, connected vehicles, healthcare etc. that are dependent on locally obtained information, require real-time analytics, or high security and privacy, benefit from cooperative computing. Cooperative computing can be combined with fog and cloud computing to address issues related to multiple applications.

D. MOTIVATION

The discussion of three main computing paradigms in IoT, i.e. cloud, fog, and cooperative computing provides two main insights. First, computing paradigms are moving from centralized to distributed computing. Second, due to the improvement of computation, storage, and communication capabilities of low-level devices, computation is being done closer to data sources. Each computing paradigm has its own virtues and drawbacks. Even cooperative which is a fully distributed paradigm, where computing is done by end devices, has some issues. A fully distributed paradigm requires high signaling overhead to coordinate devices, build and update the shared knowledge of overall network system, available resources, and received service requests [23]. The trend of computing paradigms indicates that in coming future, intelligence will be distributed and provided by end devices, however, cloud and fog computing paradigms will not vanish completely. The huge diversity in IoT devices and applications makes it almost impossible for a single computing paradigm to satisfy all application requirements. This means that these computing paradigms will have to be integrated but how this can be done is a major research challenge. This is our first main motivation to propose Edge Mesh, which aims to integrate and enable cooperation between different types of devices in the network including end devices, gateways, routers, cloud etc.

The second main motivation is related to enabling distributed intelligence in IoT. Currently, most of the existing systems do the computation and decision-making at the centralized cloud server. Even though there are few systems which try to distribute the computation, they are an exception rather than the rule. Various types of decisions are made at the cloud pertaining to resource management, device management, power management, access control, security, service management, and other application-specific processing. It is mentioned in [6] that distributed intelligence will enable right communication and computation capability at the right place. Although computation and communication are central towards enabling distributed intelligence, we should also focus on aspects such as sensing, actuation, Quality of Service, and other decision-making issues. Due to various issues associated with distributed computing systems such as synchronization, consensus, cooperation etc., it becomes challenging to enable distributed intelligence. Due to scale and complexity of IoT systems, it is challenging to determine who provides the intelligence, i.e. which functionality is provided by each device and how different devices cooperate with each other. Other issues related to device heterogeneity, data interoperability, diversity in application requirements and users, etc. makes the problem even more challenging. Applications which are critical and time-sensitive such as healthcare, autonomous vehicles, traffic management systems, security related applications, etc. require support for distributed intelligence. These applications cannot afford a delay in decision-making, which is generally the case for centralized cloud systems. Existing computing paradigms rely on the Cloud to provide intelligence and therefore, are not suitable to provide support for distributed intelligence in IoT. Edge Mesh, on the other hand, has been proposed especially to enable distributed intelligence for such type of applications.

III. EDGE MESH

In this section, we give details about the Edge Mesh computing paradigm, which aims to enable cooperation between different types of devices and enable distributed intelligence in IoT. We first give the overview of Edge Mesh paradigm. Then, we describe the overlay architecture and the proposed software framework. We also point out the specific benefits of Edge Mesh and compare it with Cloud Computing and Fog Computing paradigms.

A. OVERVIEW OF EDGE MESH

Edge Mesh can be defined as a computing paradigm that uses mesh network of Edge devices and routers to enable distributed decision-making within the network. Fig. 2 gives a high-level overview of Edge Mesh computing paradigm. A brief description of four types of devices, i.e. End devices, Edge Devices, Routers, and Cloud, shown in Fig. 2, is given below.

1) End Devices: End devices are those devices which have the capacity to sense the surrounding and change it based on the requirement. In terms of four aspects of

TABLE 1. Difference between traditional mesh networks and edge mesh.

Traditional Mesh Network	Edge Mesh
Distribute data from one node to another within the mesh network	Used for distributing data within Edge Mesh as well as enabling interaction between other devices including end devices and Cloud.
Nodes in traditional mesh network just make routing decisions	Besides routing, Edge devices in Edge Mesh make decisions for different computation tasks including processing, storage, networking etc.
Nodes in traditional mesh do not make decisions regarding end devices	Edge devices make decisions regarding interaction between end devices. Eg. Whether two devices can understand each other's data or which devices should share the data with each other
Nodes in traditional mesh are not responsible for managing data shared between end devices	Edge devices in Edge Mesh are responsible for managing data shared between end devices, which includes converting the data to required format, data aggregation, etc.

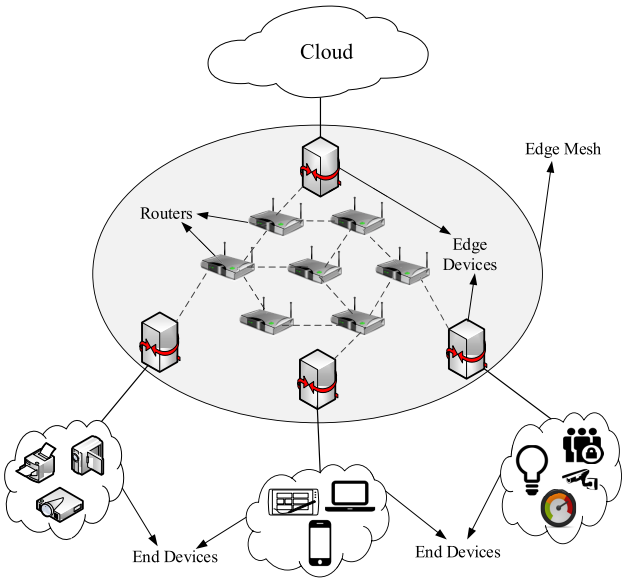


FIGURE 2. A high-level overview of edge mesh.

IoT shown in Fig. 1, End devices are responsible for sensing and actuation. Devices in a smart home such as camera, lights, thermostat, etc. are some examples of End devices.

2) Edge Devices: According to [24], Edge device is any computing or networking resource residing between data sources and Cloud-based data center. In our case, we consider Edge devices as those devices which are either connected to end devices or to Cloud. These devices are responsible for decision-making and enabling interaction between End devices. Any device that can be used for processing and enables connection between different end devices can be used as Edge device. An example of Edge device can be a smartphone, which is connected to both smart home End Devices as well as Cloud. Gateway is another example of an Edge device, which helps in connecting End devices that use different communication protocols and can also be used for various computing tasks such as processing, storage, load balancing, etc. [24].
3) Routers: Routers are used for relaying data between edge devices. Their function is just to route the data. Routers are not used for processing or enabling decision-making like Edge Devices. Routers and Edge Devices together form a mesh network which is used for sharing computation and data among Edge devices.
4) Cloud: Cloud provides abundant computing resources including networks, storage, processing, application, services, etc. Traditional IoT systems use a centralized Cloud server for enabling decision-making and other purposes as explained previously in Section II-A. However, in the case of Edge Mesh, major decision-making is done by Edge devices instead of Cloud. Cloud is integrated with other devices only to be utilized for very specific application requirements that cannot be met using Edge devices. An example of such requirement would be obtaining remote access to devices which cannot be done by use of local Edge Devices or performing big data analytics on historical data.

The architecture of Edge Mesh, shown in Fig. 2, partially looks like the architecture of wireless mesh network (WMN) shown in [25]. However, the functionalities of Edge Mesh and traditional mesh networks are quite different. A traditional mesh network is defined as a network topology in which each node relays data for the network. All mesh nodes cooperate in the distribution of data in the network. Mesh networks can relay messages using either a flooding technique or a routing technique. In mesh network topology, devices are connected using many redundant interconnections between network nodes. Table 1 outlines the differences between traditional mesh networks and Edge Mesh.

Edge Mesh integrates different devices into one computing paradigm. It incorporates characteristics from three different computing paradigms, i.e. Cloud computing, Edge Computing, and Cooperative Computing. Edge Devices are used for processing and other tasks as done in Edge Computing paradigm. Edge Devices cooperate with each other to share data and computation tasks as done in Cooperative Computing paradigm. Even though full potential of Cloud is not utilized as done in traditional IoT systems, Cloud is still a part of Edge Mesh computing paradigm and it is connected to other devices using Edge Devices as shown in Fig. 2. The second motivation of enabling distributed intelligence is addressed as all the computation tasks including processing,

storage, data sharing, and other decision-making tasks, are distributed among Edge devices.

In the case of Edge Mesh, we have considered that End devices are directly connected to Edge Devices instead of forming a mesh network with each other as done in hybrid WMNs [25]. The reason behind this architecture is that existing End devices use different communication protocols and cannot communicate with each other. For example, lights, camera, and thermostat in our homes do not interact with each other. However, we can make use of Edge devices that support conversion of data to required format to enable interaction between different End devices. Besides, the issue of interoperability between different end devices is directly related to standardization which is in the hands of standardization bodies. It will take some time to standardize the communication between End devices as there are many challenging legal and technical issues that need to be resolved [26].

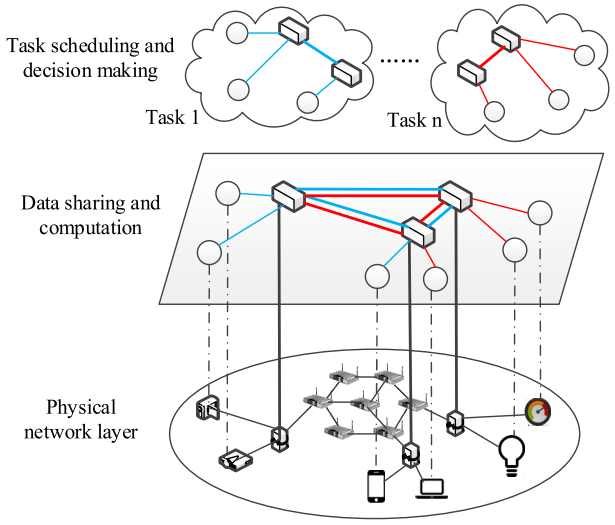


FIGURE 3. Overlay architecture of edge mesh.

B. EDGE MESH AS A COMPUTATION OVERLAY

Fig. 3 shows the architecture of Edge Mesh as a computation overlay network. At the bottom layer is the physical network that consists of routers, Edge devices, and various End devices with different computational and communication capabilities.

First, based on the physical network layer, we propose a virtual data sharing and computation layer. Routers are not shown in this layer but they provide the interconnection between different Edge devices within the Edge Mesh network. In this layer, all devices together form a virtual mesh network. Edge devices perform the computation tasks and all the devices share data by transmitting the required data to each other through the virtual mesh network. Note that different from existing paradigms in which processing happens at the servers, in our architecture, Edge devices such as gateways in the network can perform computation. That means, computation can be done locally and distributed within the network among Edge devices. Specially, as there might be several tasks running in parallel in the network, we can assign each Edge device several computation tasks and share different part of data among devices. This requires optimization of computation task assignment and data sharing schedules.

Second, on top of the data sharing and computation layer, there is an separate overlay for each task, as shown in Fig. 3. For each task, we first calculate which devices in the network should be involved, then partition the task into several computation and data sharing subtasks and assign subtasks to different Edge devices. Edge devices make decisions to partition tasks as well as determine which device should be involved. In this procedure, there are some challenging optimization issues, considering that there might already some tasks running in the network and the constraints of different devices. From the viewpoint of the task, it seems that each task runs on the network exclusively.

The main difference between Edge Mesh architecture and other computing paradigms is that Edge Mesh distributes the computation within the network instead of sending data to a server outside the network. The computation is done locally inside the network and data is shared only between the devices which are involved in the computation. The overlay architecture of Edge Mesh makes it clear that, unlike traditional mesh network where mesh routers only collect and transmit data, Edge devices in Edge Mesh are also involved in decision-making tasks.

C. SOFTWARE FRAMEWORK FOR EDGE MESH

This section discusses the software framework for Edge Mesh, which includes various functionalities implemented by different types of devices. The software framework, shown in Fig. 4, is divided into three levels. The three levels starting from bottom to top correspond to End devices, Edge Mesh, and Cloud. The details about three levels and components inside the levels are as follows:

1) END DEVICE LEVEL

This level consists of two main components, Sensing and Actuation. These two functionalities are supported by End devices. The data sensed by End devices is sent to Edge devices for processing. The processed information is then utilized by End devices to do actuation.

2) EDGE MESH LEVEL

This level is responsible for decision-making tasks and enables distributed intelligence. This level consists of two types of devices, Edge devices and Routers. Edge devices cooperate with each other to make decisions regarding sensing, communication, computation, and actuation. Edge devices also handle issues related to security, storage, both application-specific and network specific QoS requirements, etc. All computation tasks are distributed among Edge devices. Routers, on the other hand, connect different Edge devices to route the data in the Edge Mesh network.

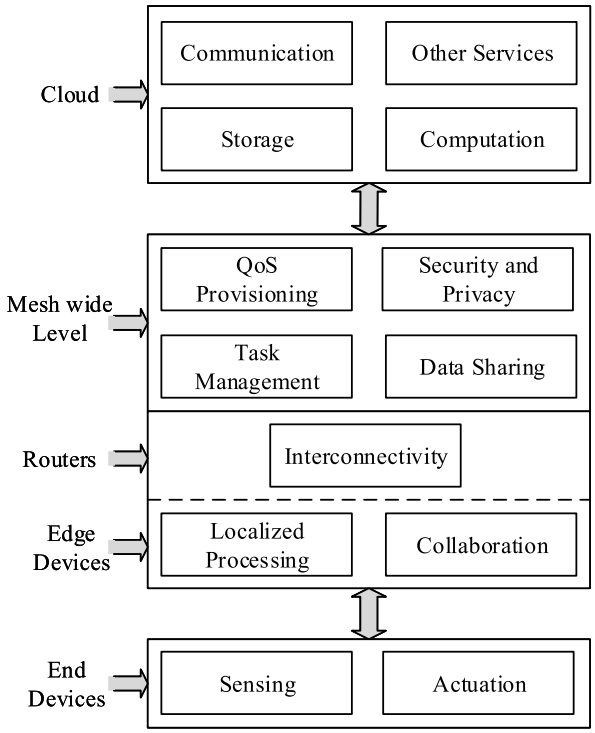


FIGURE 4. Software framework for edge mesh.

The functionality provided at this level is divided into three sub-levels corresponding to Edge devices, Routers, and Mesh-wide level functionality. Components corresponding to Edge devices are explained below.

1) Localized Processing: Edge Mesh enables processing within the network using Edge devices which do localized processing. Unlike other computing paradigms where the data is sent to a server outside the network for processing, Edge devices in Edge Mesh collect the data from End devices and distribute the decision-making tasks.
2) Collaboration: Collaboration is an important component for every distributed system. Edge devices support decision-making tasks by collaborating with each other to share data and computation tasks. Collaboration is also required to enable interaction between different End devices. Edge devices also make decisions to determine which End devices should sense the surrounding and share data with each other.
3) Interconnectivity: Interconnectivity is a common functionality for both Edge devices and routers. Interconnectivity refers to networking decisions made to exchange the data between different devices. Routers make networking decisions to route the data within the Edge Mesh network. Edge devices, on the other hand, are connected to End devices and Cloud. Edge devices make networking decisions not only to route the data with the Edge Mesh network but also to connect different End devices and exchange the data with the Cloud.

The three functionalities discussed above belong to local-level decision-making. However, Edge devices work together to support global level functionalities which include global sharing of data and knowledge, task distribution and scheduling, and other high level decisions with respect to QoS, Security, and Privacy. We have grouped these functionalities under Mesh-wide level. The components corresponding to Mesh-wide level are:

1) Task Management: Edge Mesh divides complicated computation tasks into sub-tasks which are distributed among different Edge devices. This component is responsible for making decisions with respect to task distribution and scheduling. A challenging issue here is to develop lightweight distributed algorithms for task management. We have discussed in detail about task management and specifically the task allocation problem in Section IV.
2) Data Sharing: This component is responsible for sharing data between different Edge devices using the Edge Mesh network. Data sharing also involves decision-making for other aspects related to data management such as data aggregation, data conversion etc. The data generated and processed by different devices in the network is shared among devices. A single device generates and processes vast amount of data which is to be shared selectively among devices. It is important to determine which data is shared and how is it being shared. Simply broadcasting all the data leads to network congestion and data redundancy. Other issues such as data relevancy must are also be handled by this component. For example, in intelligent transportation systems, the data is only valid within a small space and for a short period of time. Data sharing also involves data conversion to the required format to resolve interoperability issues. Interoperability issue can arise due to difference in communication protocol or it can be a semantic interoperability issue. Data sharing component makes sure that data is not only in the right format but also the meaning of data can be understood by devices.
3) QoS Provisioning: Decisions related to different computation tasks including processing, networking, data sharing, etc. are influenced by network-specific and application-specific QoS requirements. Edge devices make decisions to consider different QoS factors such as latency, reliability, availability, etc. However, different QoS factors usually contradict with each other leading to different system configurations. Therefore, Edge devices are responsible for making trade-off decisions to accommodate different requirements at Mesh-wide level.
4) Security and Privacy: Data and computation tasks are shared at mesh-wide level in Edge Mesh which leads to security and privacy concern. This component makes sure that data can only be accessed by authorized entities. Other security issues such

TABLE 2. Comparison between cloud computing, fog computing, and edge mesh.

Characteristics	Cloud Computing	Fog Computing	Edge Mesh
Response Time	Minutes to weeks	Milliseconds to Minutes	Milliseconds to Minutes (Equal or better response time than Fog Computing)
Information type	Global Information that is highly abstracted	Limited localized information	Limited localized information but it is shared among Edge Devices
Data analytics	Big Data analytics	Simple analytics, visualization	Higher analytics capability than simple analytics due to sharing of computation
Geographical coverage	Global	Local	Local
Hardware resources	Ample computation resources and storage space	Limited computation resources and storage space	Limited computation resources and storage space but they are shared
Load distribution	No	Partially	Yes
Scalability	Low	Higher than Cloud Computing	Higher than Fog Computing
Deployment	Centralized	Centralized or Semi-distributed	Distributed

as authorization, protocol and network security, and security against attacks are handled by this component.

3) CLOUD LEVEL

Although majority of decision-making is done at Edge Mesh level, Edge Mesh computing paradigm includes Cloud as it can provide abundant resources including storage, communication, computation, development tools, and other services. Cloud can enable IoT application users to connect, monitor, or control any device from any place, assuming it has Internet connectivity [13]. Low-cost and easy communication is enabled by the integration of Cloud with IoT. Cloud resources are also used for Big Data analytics.

D. BENEFITS OF EDGE MESH

The main objective of Edge Mesh is to enable distributed intelligence which helps Edge Mesh to provide benefits associated with distributed computing systems. Such benefits include fault tolerance, better scalability, and efficient performance due to the distribution of load. There are some other benefits provided by Edge Mesh by the virtue that it integrates characteristics from three different computing paradigms, i.e. Cloud Computing Fog Computing, and Cooperative Computing. Edge Mesh provides the best features of the three computing paradigms. The benefits provided by such integration include low latency, better services, and higher security and privacy.

Table 2 shows a comparison between Cloud Computing, Fog Computing, and Edge Mesh. The features selected for comparison have been taken from [22] and a white paper by Cisco [27]. Edge Mesh is very different from Cloud computing but it is similar to Fog Computing in some ways. Both Fog computing and Edge Mesh focus on moving data processing closer to data sources. However, Edge Mesh focuses on cooperation between Edge devices to share data and computation with each other. Features like load distribution and cooperation are focused in Edge Mesh but they have not been clearly defined in Fog Computing. Sharing of resources among Edge Devices leads to better response time, higher analytics capability, better services, etc. A detailed discussion about all the benefits provided by Edge Mesh and their significance have been given below.

1) Fault Tolerance: Edge Mesh provides fault tolerance in terms of both communication and computation. Since a mesh network is used for distributing data among different devices, it provides many redundant connections. In the case of failure of a device in the communication path, other paths can be used for distributing data. Edge Mesh also provides redundancy for computation tasks. The responsibility of any computation task lies on multiple Edge Devices that cooperate with each other, therefore, failure of a single device does not jeopardize the whole system. Edge Mesh is useful for critical applications such as healthcare, traffic light control, emergency warning systems, etc. that require high reliability.
2) Scalability: The high-level overview of Edge Mesh shown in Fig. 2 shows a hierarchical network structure which is suitable for scaling. Scalability is an important requirement for IoT systems as the number of devices will continue to increase in the coming future. A computing paradigm that relies on the centralized server for computation tasks cannot be scaled up. Edge Mesh, on the other hand, has been proposed to enable distributed intelligence which makes it suitable for IoT applications. A major challenge to scalability is communication bottleneck due to limited bandwidth in IoT systems. However, Edge Mesh is distributed so all the data is not sent to a single Edge device. The data is sent to multiple Edge devices which can then share data so the communication bottleneck issue is resolved due to the distributed nature of the system.

3) Load Distribution: Computation tasks can be offloaded to other Edge devices which speed up the processing time. A single Edge device is not overloaded which usually leads to better performance. Edge Mesh distributes the load among Edge devices which leads to better response time, reduced makespan, and higher throughput. Distribution of load also makes the systems more flexible, i.e. in the case of device failure, other devices can share the load of failed device. IoT systems are dynamic, as devices can be mobile, added, removed, or changed in configuration. Edge Mesh can adjust to such changes as Edge devices can cooperate with each other. For example: if a complex task cannot be handled by current device, the device can either offload some of the task components to other devices or even fully offload the task to a better device which can handle the task.
4) Low Latency: Many IoT applications such as healthcare, video analytics, autonomous vehicles, traffic management systems, emergency response systems, smart parking, etc. have low latency requirement. Cloud computing paradigm is not efficient enough to be used for these time-critical applications. A large portion of the time is consumed to transfer the data to and from a remote server which does all processing tasks. Edge Mesh uses local Edge devices which can perform computation tasks and share data within the required deadline. This aspect is common with Edge Computing where computation is also done closer to where data is generated and consumed. In the case of Edge Mesh, Edge devices also cooperate with each other to provide even better response time.
5) Better services: End devices contain sensors that generate data which is then processed to create useful information and provide different services. For example: Nest thermostat uses machines learning algorithms and sensing technology to generate personalized heating and cooling schedule [28]. The type of knowledge generated and the service provided depends directly on what kind of data is being used. In the case of Cloud computing, complete raw data is not sent to the centralized server due to bandwidth limitation, therefore, only limited services can be provided based on the abstracted data that is sent. Edge Mesh uses local Edge devices which can make use of complete information and have context awareness which can help in generating better useful information and thus, provide better services. For example, Edge Mesh can be used in Smart Home application to learn detailed information about human behavior and activities, which can help in providing better services. Many other data-driven IoT application such as manufacturing, energy management, healthcare, etc. can benefit from distributed data analytics enabled by Edge Mesh [7].
6) Local Processing: In Edge Mesh, data processing is done by Edge devices which are located locally. The data is not sent every time to some remote server for analytics. The data is only shared among those devices which need the data for processing. The communication with outside unknown entities is minimized. Edge Mesh also enables provision of services even in the case of Internet failure as data analytics is done locally. This is useful in case of emergency rescue operations where Internet services are usually not working and it is important to do the processing locally and within a shorter time duration.
7) Better Security and Privacy: Security and Privacy are an important concern as a vast amount of data is collected which can be used to identify personal information. This data can fall into the hands of wrong people who can use it to hamper our lives. Edge Mesh leads to be better security and privacy as it uses local processing. Data is not shared with outside entities and no communication is required with intermediary nodes which are usually prone to security attacks. Since whole data is not shared using the Internet, it cannot be accessed easily by anyone. Although it should be noted that Edge Mesh will require distributed security and privacy algorithms which are challenging to develop as compared to centralized ones.

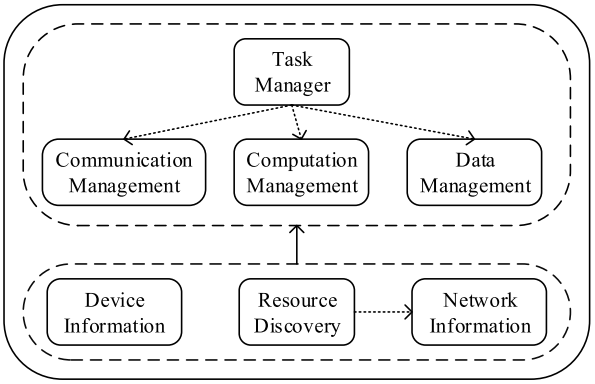


FIGURE 5. Task management framework.

IV. TASK MANAGEMENT FRAMEWORK

This section discusses the task management framework and more specifically the task allocation problem in Edge Mesh. Task management framework, shown in Fig. 5, is used for managing and distributing tasks among Edge Devices. The task management framework is divided into two parts, top and bottom. The top part contains task manager and other components that help in the management of tasks. Bottom part contains components that store and retrieve the information about the network and other resources. This information is then utilized by components in the top part to manage the tasks. The details about various components within the task management framework are given below.

1) Task Manager: Task manager is the main component which moderates other components in the framework to decide how tasks are allocated to each device.

Multiple tasks are usually dependent on each other, therefore, task manager must decide not only where the tasks are allocated but also when the tasks are executed. Tasks are allocated based on many factors including resources accessibility, network structure, task objectives, QoS requirement for the application, load balancing, etc. Task allocation can be done based on many objectives such as minimizing makespan, maximizing throughput, minimizing energy consumption, minimizing communication, etc.

2) Computation Management: Computation management component makes decisions regarding task processing and computation sharing. This component decides determines the computational need for a task and allocates the tasks based on resources present on each device and task objectives. One of the main objectives of computation management is to share the load among different Edge Devices. Load balancing is not just an objective but it is also a technique to optimize other objectives including response time, makespan, and throughput [29].
3) Communication Management: Communication is a major contributor towards resource consumption of embedded devices used in IoT. Task management in Edge Devices involves joint optimization of computation and communication, therefore, communication management component plays a major role. Communication management component handles routing of data within the Edge Mesh. This component is responsible for determining which devices can communicate with each other and what should be the best path to share data between different devices.
4) Data Management: Task Management decisions are influenced by resource accessibility. Data Management component makes decisions regarding which devices should share the data depending on where the data is located and resources consumed in accessing the data. This component works together with communication management component to determine the best path for sharing data.
5) Resource Discovery: Resource discovery implies discovery of devices which can provide the required information or service for any task. It is challenging to reliably discover resources in a reasonable amount of time as the network is huge and could be disconnected. The information obtained by this component is utilized to determine which devices should interact to share the data and computation. The decisions made by task manager are dependent on the information discovered by this component.
6) Device Information: This component stores all the information relevant to a device such as types of sensors on the device, metadata on information generated by the device, metadata on tasks that are currently running on the device, existing usage of resources including power, memory, CPU of the device. Tasks allocated to each Edge device are dependent on the information provided by this component. This information is utilized by computation management component and data management component to make decisions regarding sharing of computation and data, respectively among devices.
7) Network Information: This component stores all the information about routers, neighbouring Edge devices, connected End devices, different sensors in the network, etc. The information collected by resource discovery component is transferred and stored in this component. Routing decisions made by the communication management component are done based on the information stored in this component.

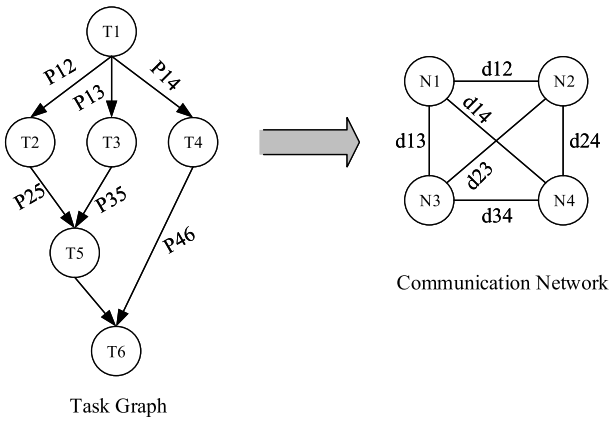


FIGURE 6. Task allocation in edge mesh.

A. TASK ALLOCATION IN EDGE MESH

This section describes the preliminary study done on task allocation problem in Edge Mesh. Task allocation problem, shown in Fig. 6 for 6 tasks and 4 devices, is to allocate a set of dependent tasks to a set of Edge Devices such that total energy consumption is minimized, and energy consumed by each device is less than given initial energy. Edge devices in Edge Mesh are responsible for task allocation and decision-making. Task Allocation problem has been studied for long time for WSN [30], IoT [31], etc. However, a major difference in case of Edge Mesh is that input data for different tasks are distributed as data is generated by sensing devices which are located at different geographical places. There are very few existing works which also consider data distribution for task allocation and scheduling problem [32], [33]. To the best of our knowledge, there is no existing work which considers data distribution, task dependency, embedded device constraint, and device heterogeneity simultaneously for task allocation problem as considered in our problem.

1) PROBLEM FORMULATION

This section describes the modeling of application, network, data, and cost functions. The problem is formulated as a non-linear integer programming problem. Mathematical notations used in the problem formulation are summarized in Table 3.

TABLE 3. Mathematical notations used in problem formulation.

Variable	Meaning
t_i	i^{th} task
p_i	computation load of processing task i
P_{ij}	length of data in bits transferred between tasks t_i and t_j
a_j	j^{th} device
l_{ij}	length of data in bits transferred from device j for executing task t_i
M	number of tasks
N	number of devices
X	Matrix of task allocation
x_{ij}	variable to determine whether task t_i is allocated on node a_j 1 for yes, otherwise 0
e_j	average consumption power of device a_j (J/sec)
v_j	processing speed of node a_j
$e_{j,init}$	initial energy capacity of node a_j
d_{jk}	distance between node a_j and a_k
R_i	number of predecessor tasks for task t_i
S_i	number of successor tasks for task t_i
K_i	number of devices where input data required for task t_i is distributed
W_j	set of tasks which require data from node a_j
$\in_{elec}$	energy consumption of operating the device of each bit of data
$\in_{amp}$	coefficient of transmit amplifier
$\|j-k\|$	variable to determine whether a_j and a_k are same devices. Its values is 0 if they are same and 1 otherwise
$\|x_{ij}-x_{rj}\|$	variable to determine whether tasks t_i and t_r are executed on same node a_j. 0 for same, 1 otherwise
E_{ij}	total energy consumption for executing task t_i on node a_j
E_j	Total energy consumed by node a_j for all tasks

a: APPLICATION MODEL

The application is modeled as a directed acyclic graph (DAG) G = (T, P), where T is the set of tasks, T = $\{t_1, t_2, \ldots, t_m\}$, and P is the set of dependencies between the tasks. Number of tasks in task graph is M. Each task t_i has a computation load of processing, p_i. Weight of each link connecting tasks t_i and t_j is P_{ij}, which represents the amount of data to be transmitted if the tasks are executed on different devices. Task t_i has some predecessor tasks which is given by a set R_i. Set of successor tasks is given by set S_i.

b: NETWORK MODEL

The communication network is a mesh network of Edge devices connected to each other. The communication network is modeled as a graph N = (A, D), where A is the set of devices, A = $\{a_1, a_2, \ldots, a_n\}$, D = d_{ij} represents the distance between device a_i and a_j. Number of devices in communication network is N. Each device a_i is heterogeneous in terms of processing power and battery capacity. For each device a_i, $\{v_j, e_j, e_{j,init}\}$ gives the initial list of parameters, where v_j is processing speed, e_j is the average power consumption, and $e_{j,init}$ is the initial energy capacity of device a_j.

c: DATA MODEL

Input data for tasks is distributed among devices. L is a MxN input data matrix. Each element l_{ij} is number of bits required for task t_i from node a_j. Data for task t_i is distributed among K_i nodes in the network. Each node a_j provides data for W_j tasks.

d: TASK ALLOCATION MATRIX

X is a MxN order matrix. Each element x_{ij} in the matrix represents whether task t_i is executed on node a_j. Value of x_{ij} is 1 if task t_i is executed on node a_j, 0 otherwise.

1) **Total Cost for processing a task t_i on node a_j, E_{ij}**: The total cost is the sum of computation cost, cost to communicate input data, and cost to communicate data from predecessor tasks.

a) *Cost for processing a task t_i on node a_j*

$$E_{comp} = e_j * \frac{p_i}{v_j} \tag{1}$$

b) *Cost for communicating data for task t_i executed on node a_j*

$$E^t_{comm,d} = \sum_{1 \le k \le K_i} (((\in_{elec} + \in_{amp} * d^2_{jk}) * l_{ik}) * (|j-k|)) \tag{2}$$

$$E^r_{comm,d} = \sum_{1 \le k \le K_i} (((\in_{elec} * l_{ik}) * (|j-k|)) \tag{3}$$

$$E_{comm,d} = E^t_{comm,d} + E^r_{comm,d} \tag{4}$$

where, $E^t_{comm,d}$ is the transmission cost, $E^r_{comm,d}$ is the receiving cost, and $E_{comm,d}$ is the total cost.

c) *Cost for communicating data for task t_i from its predecessor tasks*

$$E^t_{comm,t} = \sum_{1 \le r \le R_i} (((\in_{elec} + \in_{amp} * d^2_{jr}) * P_{ri}) * (|x_{ij} - x_{rj}|)) \tag{5}$$

$$E^r_{comm,t} = \sum_{1 \le r \le R_i} (((\in_{elec} * P_{ri}) * (|x_{ij} - x_{rj}|)) \tag{6}$$

$$E_{comm,t} = E^t_{comm,t} + E^r_{comm,t} \tag{7}$$

where, $E^t_{comm,t}$ is the transmission cost, $E^r_{comm,t}$ is the receiving cost, and $E_{comm,t}$ is the total cost.

d) *Total Cost of executing task t_i on node a_j*

$$E_{ij} = E_{comp} + E_{comm,d} + E_{comm,t} \tag{8}$$

2) **Total Cost of Energy Consumption for node a_j , E_j**: Each device consumes energy for processing the tasks as well communicating data. Different costs including processing cost, communication cost for transferring input data or data between tasks have been explained below.

a) *Cost of processing task t_i*

$$E_{j,comp} = e_j * \frac{p_i}{v_j} \tag{9}$$

b) *Cost of sending data between task t_i and its successor tasks S_i*

$$E^t_{j,comm,t} = \sum_{1 \le s \le S_i} (((\in_{elec} + \in_{amp} * d^2_{js}) * P_{is}) * (|x_{ij} - x_{sj}|)) \tag{10}$$

c) *Cost of sending input data for task w belonging set of W_j tasks executed on some node a_q*

$$E^t_{j,comm,d} = \sum_{1 \le w \le W_j} ((\in_{elec} + \in_{amp} * d^2_{jq}) * l_{jw} * x_{wq}) \tag{11}$$

d) *Cost of receiving input data from other nodes for task t_i executed on node a_j*

$$E^r_{jcomm,d} = \sum_{1 \le k \le K_i} (((\in_{elec} * l_{ik}) * (|j-k|)) \quad (12)$$

e) *Cost of receiving data from predecessor tasks R_i for tasks t_i executed on node a_j*

$$E^r_{j,comm,t} = \sum_{1 \le r \le R_i} (((\in_{elec} * P_{ri}) * (|x_{ij} - x_{rj}|))) \quad (13)$$

f) *Total energy cost on node a_j for all tasks*

$$E_j = \sum_i (E_{j,comp} + E^t_{j,comm,t} + E^r_{j,comm,t} + E^r_{j,comm,d}) * x_{ij} + E^t_{j,comm,d} \quad (14)$$

3) **Optimization Problem**: The optimization problem is formulated as:

Objective

$$minimize \quad (\sum_i \sum_j E_{ij} * x_{ij}) \quad (15)$$

Constraints:

$$\sum_j x_{ij} = 1 \quad (16)$$

$$E_j \le e_{j,init} \quad (17)$$

$$x_{ij} = 0 \text{ or } 1 \quad (18)$$

where, x_{ij} is the main variable of this problem, i $\in$ {1, 2, …, M}, and j $\in$ {1, 2, …, N}. The objective is to minimize the total energy consumed by all tasks (15). The constraints are that each task is executed on exactly one device (16), total energy consumed by each device is less than its initial energy (17), and the variable x_{ij} can take value either 1 or 0 (18).

2) PROPOSED SOLUTION AND EVALUATION

We have proposed a genetic algorithm to solve the task allocation problem as shown in Algorithm 1. The algorithm is similar to one proposed in [34], however we have changed the crossover and mutation operator.

The input variables for the proposed genetic algorithm are GGAP (generation gap used for crossover operation), MUTR(mutation rate), ITER(number of iterations for the algorithm), and NINM(number of members in the population). The algorithm starts by creating random population (line 1) which are task allocation matrices. The rows in task allocation matrix are arranged from top to bottom based on the task precedence order. So if a task is before another task in task graph then row corresponding to that task is above. In case multiple tasks have same level. Then row corresponding to these tasks are arranged randomly. Each column of the task allocation matrix corresponds to the device. For each row, only 1 column has value 1, rest others are 0. Cost and Fitness values of each member of the population is evaluated (lines 4 and 5). We convert Cost value to fitness

Algorithm 1 Proposed genetic algorithm

Input: GGAP, MUTR, ITER, NINM
Output: Best Task Allocation

```
1  Chrom ← RandomlyCreatePopulation(NINM);
2  Gen ← 0;
3  while Gen < ITER do
4      CostValue ← Costfunction(Chrom);
5      FitValue ← ConvertToFitness(CostValue);
6      Select ←
       RouletteWheelSelect(Chrom, FitValue);
7      i ← 1;
8      while i < GGAP * NINM do
9          CrossOver(Select(i), Select(i + 1));
10         i ← i + 2;
11     end
12     for i ← 1 to GGAP * NINM do
13         Select(i) ← Mutate(Select(i), MUTR);
14     end
15     Chrom ← Reinsert(Chrom, Select, NINM);
16     Gen ← Gen + 1;
17 end
18 CostValue ← Costfunction(Chrom);
19 i ← GetIndexOfMinimum(CostValue);
20 return Chrom(i);
```

value by taking its inverse. Cost is calculated using following equation

$$Cost = (\sum_i \sum_j E_{ij} * x_{ij}) - H * (\sum_j min(0, e_{j,init} - E_j)) \quad (19)$$

where, H is some very large number.

A percentage of initial population, GGAP, is selected for crossover operation. The selection is done using Roulette Wheel Selection. In lines 8-10, GGAP*NINM members are selected and crossover operation is performed. An example of crossover operation involving two 4×4 matrices is shown below. The position of line is chosen randomly.

Before Crossover Operation

$$\left[\begin{array}{cccc} a & b & c & d \\ \hline e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{array}\right] \left[\begin{array}{cccc} A & B & C & D \\ E & F & G & H \\ I & H & K & L \\ \hline M & N & O & P \end{array}\right]$$

After Crossover Operation

$$\left[\begin{array}{cccc} a & b & c & d \\ \hline A & B & C & D \\ E & F & G & H \\ I & H & K & L \end{array}\right] \left[\begin{array}{cccc} e & f & g & h \\ i & j & k & l \\ m & n & o & p \\ \hline M & N & O & P \end{array}\right]$$

Mutation is then performed on the members obtained after crossover operation (lines 12-14). Mutation operation involves swapping a column element having value 1 with other random column element for each row. New obtained population is combined with the initial population and members with the least cost value equal to the size of initial population are selected (line 15). This process is repeated for ITER iterations. After all the iterations are completed,

cost value of each member of final population is evaluated and member with the minimum cost value is returned (lines 18-20)

We have implemented the proposed genetic algorithm in MATLAB and done some preliminary simulation experiments by using random parameters used previously in [30]. The parameters are set as: bandwidth = 250Kbps; e_{elec} = 50nJ/b; e_{elec} = 10pJ/b/m^2; initial energy to be uniform distribution in the range [10, 20] mJ; processing speed to be uniform distribution in the range [30, 100] MCPS; power consumption to be uniform distribution in the range [4, 10] mW; computation load to be uniform distribution in the range [300, 600] KCPS; communication load for each task to be uniform distribution in the range [500, 800] bytes of data; distance between devices to be uniform distribution in the range [1,500] meters. For the input data matrix, the input data for each task is distributed randomly among less than half of the total number of devices. The number of bytes sent from each input data source is set to be uniform distribution in the range [300, 1000] bytes. The parameters for genetic algorithm are GGAP = 0.8, mutation rate = 0.04, size of initial population = 30, number of iterations = 50, and H = 10^7.

TABLE 4. Cost Value for different number of devices.

Number of Devices	Mean Cost Value	Median Cost Value
4	0.0029	0.0025
6	0.0040	0.0037
8	0.0054	0.0052
10	0.0072	0.0069

The simulation experiments have been performed by using the task graph shown in Fig. 6, and varying the number devices. Table 4 shows the average cost value after running the experiment 100 times. The initial results show that proposed algorithm is able to achieve good results, however, more extensive experiments are required to test its efficacy. Another observation is that as the number of devices in communication network are increased, the cost value also increases. This is expected as number of input data sources for each task is equal to half the number of total devices. Therefore, extra cost associated with data transfer is added to total cost. The proposed algorithm also converges faster, however, more experiments are required to give an inference.

This work needs to be extended further to achieve better results and have accurate validation. We have only proposed one genetic algorithm and not compared it with other algorithms. Besides, more extensive experiments are required under different parameter settings to have a better validation of the proposed approach. The problem considered in this work has been simplified as we have considered fully connected communication network consisting of only Edge devices, however, communication in Edge Mesh is done using multi-hop path and Edge Mesh consists of other devices too. Besides, the objective of task allocation problem considered here is to minimize total energy consumption but real world applications have other objectives too such as minimizing latency, maximizing reliability, etc.

V. APPLICATION SCENARIOS

The benefits of Edge Mesh such as distributed processing, fault tolerance, low latency, etc. have been discussed in Section III-D. This section presents some application scenarios which require these features. The application scenarios discussed in this section are related to three different application domains, Smart Home, Intelligent Transportation System, and Healthcare. These application scenarios help in illustrating the benefits of Edge Mesh and give an understanding of scenarios where Edge Mesh computing paradigm can be of significant use.

A. SMART HOME AND BUILDING

Smart Home has been one of the oldest application domain of Internet of Things. The main objective of Smart Home is to improve the comfort level, security, and safety of people inside the home while considering energy conservation and cost into account. A recent research trend is to enable cognitive capacity in Smart Homes [35]. The incorporation of cognitive capability in smart homes requires devices to coordinate with each other to provide a wide range of services such as the autonomous adaptation of HVAC systems, lighting, etc. The services provided for Smart Home can be classified into four types, i.e. Comfort and Convenience, Energy Conservation, Security and Safety, and Heath Care. We give two different scenarios in Smart Home where Edge Mesh can be used to enable distributed intelligence.

The first scenario is the falling-asleep problem, which has been previously discussed in [35] and [36]. This scenario belongs to comfort and convenience category. In this scenario, it is imagined that a person who is sitting on a sofa and watching TV, gradually falls asleep. The problem here is to change the surroundings accordingly such that person can get a comfortable sleep. The changes include dimming the lights, reducing TV volume and switching it off, dynamically changing the air-conditioner setting based on body and room temperature, slowly changing the sofa into the shape of the bed, etc. These changes require different types of devices such as lights, TV, sofa, wearables, to share data and coordinate with each other. There are many sub-problems here such as detecting if the person is sleeping, making changes to the surroundings without disturbing the sleep, etc. This problem requires the use of complicated algorithms that cannot be executed on a single device but as proposed in Edge Mesh, we can distribute the processing load among different devices. The whole application can be divided into a set of tasks which can be distributed among different devices using the proposed software framework. The whole decision-making can also be done at a centralized server but that would require transferring the whole data which would take longer time and pose security concerns as data would be shared with devices

that are outside the smart home network. Besides, Edge Mesh would be able to achieve a better result in changing the surroundings as devices have access to whole raw data.

The second scenario is the emergency building evacuation in case of fire. This scenario belongs to the security and safety category. Fire emergency building evacuation is a critical safety application which requires high reliability and faster response time. Due to damage in infrastructure, Internet service can be down which makes it difficult to communicate with outside entities and provide efficient evacuation service. Existing systems use centralized server for making decisions regarding alarm generation and building evacuation [37], [38]. Building evacuation, in the case of fire, requires input from many different devices to generate an alarm, detect occupants, and guide them to the nearest exit. Alarm system further requires input from smoke sensors, heat sensors, light sensors to reliably detect location and intensity of the fire. Building management system in [37] stores data about the number of people, their age and disability status, and occupancy status of each room, that can be used to provide a dedicated plan for each floor and person. The data should be shared with different devices within the system in case of fire emergency. It is possible that one part of the building may be severely damaged due to fire, so an alternative building evacuation strategy should be provided which requires coordination between different systems and devices. The devices should not only share data but also make decisions related to building evaluation. The information is also shared with rescue workers to guide them and use their feedback to dynamically change the building evacuation plan. Such system requires very low latency and this can be provided by using Edge Mesh computing paradigm. The whole application can be divided into sub-tasks such as alarm generation, occupancy detection, localization, evacuation plan, etc. which can be distributed among different Edge devices that share data as well computation. Compared to a centralized system, Edge Mesh has the potential to provide faster, reliable, and efficient building evacuation service.

Besides the two scenarios, there are many other scenarios that can use Edge Mesh such as energy management, ambient intelligence, etc. Energy management requires dynamic scheduling of devices as demand for energy can change based on time and condition. Even the cost can change dynamically due to peak-demand charges, or time-of-use tariffs [39]. A few case studies related to ambient intelligence in Smart Home have been given in [36].

B. INTELLIGENT TRANSPORTATION SYSTEMS

Intelligent transportation system (ITS), also referred to as Internet of Vehicles (IoV), is another important application domain of IoT. The technological advancement in sensor technologies and vehicular communication technologies such as Wireless Access for Vehicular Environments (WAVE), which is a communication protocol to enable data exchange between high-speed vehicles and between vehicles and roadside infrastructure units, has enabled exciting applications for ITS domain. Google and other big corporations are now working on autonomous and connected vehicles to improve road safety, traffic efficiency, and enable other services such as intelligent parking, accident prevention, collision warning, etc. New vehicles are now being equipped with many sensors and on-board navigation units to help improve the safety of drivers and make the driving experience more comfortable. Vehicles can now be connected to the Internet and with each other to allow sharing of data about road and traffic conditions. Integration of vehicles, smartphones, wearables, and other sensors can lead to many business opportunities too such as entertainment services, advertisement, etc.

ITS applications require some specific features such as high and constrained mobility, real-time processing, spatial and temporal dependence of data, etc. A centralized computing solution cannot be efficiently used for ITS applications due to these features. If a centralized computing solution is used then every vehicle will have to send a large amount of data using the Internet to a server which is usually located someplace far away. Since vehicles travel at high speed, data generated by vehicles is only valid for a short period of time and within a small distance. If the number of vehicles is very high, it will also create communication bottleneck which further leads to more delay. Besides, if the data generated by vehicles such as real-time location, and origin and destination information is shared using the Internet to a centralized server, user identity can be inferred which leads to privacy concerns [40]. Edge Mesh is a suitable computing paradigm for such applications as it provides benefits such as low latency, distributed processing, better security and privacy, etc., which cannot be provided by centralized computing solutions. Edge Mesh is especially useful for ITS application scenarios which require coordination among vehicles. Applications such as autonomous vehicles, traffic light control, managing evacuation, etc. fall under this category. Traffic management scenario has been discussed below to illustrate the benefit of Edge Mesh for ITS applications.

Traffic management system includes congestion detection and congestion avoidance, but it is also related with other management systems such as traffic light management and parking management. Re-routing of traffic for congestion avoidance requires sharing of information among vehicle from different road segments. Vehicles should share information to determine alternative routes but here again, all the vehicles cannot be routed to same alternative route as it would only result in shifting the congestion from one road to another. Vehicles must coordinate with each other so that congestion does not oscillate from one road segment to another. A survey of traffic management systems using wireless sensor networks has been done in [41]. Most of the solutions proposed in literature for congestion detection and avoidance use a centralized server. Recently, a fully distributed traffic management system was proposed in [42], however, the solution proposed in [42] focuses on congestion

detection and does not give detailed strategy for congestion avoidance. A hybrid solution for congestion avoidance has been proposed in [40]. The algorithm proposed in [40] is based on centralized algorithm previously proposed in [43]. While implementing a distributed solution using Edge Mesh, each vehicle consisting of various sensors can be considered as an Edge device. A major challenge in developing a solution for ITS application is to determine how the devices should interact and share data. As discussed previously for Smart Home application, a complicated application can be divided into sub-tasks which can be distributed among Edge Devices. However, it is not trivial to propose a fully distributed solution for traffic management application due to many data management challenges such as synchronization, determining the relevance of information, disseminating the information efficiently, data aggregation, avoiding broadcast storm, etc.

C. HEALTHCARE

Healthcare is another popular application of IoT. IoT offers the potential for many healthcare applications such as glucose level sensing, blood pressure monitoring, body temperature monitoring, medication management, rehabilitation system, wheelchair management, etc. [44]. Common features associated with these applications is that most of them require real-time processing, high reliability, high accuracy, mobility support, and high security and privacy. As discussed before, centralized computing solutions cannot support these features. Edge Mesh, on the other hand, is proposed to enable distributed intelligence and can provide support for these features. We discuss the scenario of how IoT healthcare can be used to save patient's life and why Edge Mesh is a suitable computing paradigm for such scenarios.

A pilot study was conducted in [45] to capture in-home activity data including meal plan, hygiene, movement, blood pressure, etc. using sensors. The study did not involve real-time analysis of data. During the pilot study, a user who was an elderly person died from a heart stroke that occurred during the night. The authors who conducted the study analyzed the data later and found some changes in user activities which could have given an indication of impending heart stroke. It was found that the user had reduced mobility, reduced meal preparation, reduced hygiene, high blood pressure, fluctuating blood pressure, low activity, and loss in weight [45]. These changes in activities are not of much use when considered alone but they can point to an impending heart stroke when considered together. The main point behind this is that data collected from different activities need to be shared and devices need to coordinate with each other to give a useful information. Decision-making cannot be done based on a single source of data. Real-time processing is important to enable timely response in case of emergency. Edge Mesh is helpful in these scenarios as it can enable distributed analytics and sharing of data to generate useful information and timely response.

VI. OPEN CHALLENGES

The two main characteristics of Edge Mesh are that it uses distributed computation and integrates characteristics from different computing paradigms. These features result in many benefits as discussed in Section III-D, however, they are also the reasons behind many research challenges in implementing Edge Mesh. Edge Mesh should support communication between different types of devices such as Edge device, End device, routers, and Cloud. The data should not only be shared between different types of devices but must be understood by the devices. Communication protocols used in IoT suffer from low data rate, frequent packet losses, and stochastic channel variations, which make it difficult to achieve reliable communication [26]. Edge Mesh enables distributed intelligence, however, where the intelligence should be placed is a major research question. There are many other research questions too including, How the devices cooperate with each other? Which devices should share data? How do devices decide which data to be shared? Who decides the distribution of tasks among devices? Which factors determine the distribution of tasks? etc. The computation tasks are usually distributed among Edge devices, but Cloud can also be used for big data analytics on large historical data. So, the tasks can be distributed among different devices depending on application requirement and resources available on the devices. It is challenging to determine how the tasks must be distributed among Edge devices as it requires joint optimization of computation and communication. Edge devices need to take care of many issues including access control, resource allocation, QoS, security, data conversion, data management, etc. This requires Edge devices to be robust and flexible. It is challenging to manage all the tasks simultaneously using Edge devices as these devices are heterogeneous, resource-constraint, and distributed. The challenges in implementing Edge Mesh are a combination of many factors including wireless distributed computing issues, IoT related challenges, embedded device constraints, software implementation issues, theoretical modelling limitations, algorithmic challenges, issues related to distributed data analytics, etc.

Edge Mesh uses wireless distributed computation for various computation tasks and network management. Wireless distributed computing poses many challenges that are different from challenges in traditional distributed computing systems as discussed in [46]. Distributed algorithms from traditional distributed systems cannot be directly applied for Edge Mesh due to many differences. Edge Mesh is a dynamic network as devices can be mobile, heterogeneous, reconfigured, replaced, or even randomly fail. The wireless channel also suffers from stochastic variations leading to intermittent connectivity and other uncertainties that make the whole system difficult to model. Theoretical modeling of application and network is important to allocate tasks and evaluate performance. Besides, the devices have limited resources such as memory, energy, computation, and communication capability, which further makes the system design

more challenging than traditional distributed systems. Edge Mesh can have multiple tasks running simultaneously and these tasks can also be shared between different applications. Due to heterogeneity in device capabilities, the type of task supported by a device can vary and sometimes a task may not be supported by a device due to its limited resources. Such issues make the problem of resource allocation and scheduling more challenging. Other fundamental issues related to wireless distributed computing systems include routing, process and clock synchronization, leader election, topology control, mutual exclusion, network information management, etc.

Distributed data management and analytics also pose many research challenges in implementing Edge Mesh. One important challenge related to distributed data storage is, How the network information is distributed among devices to enable operations such as efficient resource discovery, failure recovery, stability analysis, etc.? [46]. The data analytics algorithms implemented for centralized systems or traditional distributed systems cannot be directly applied for Edge Mesh. Edge Mesh requires implementation of new lightweight algorithms for resource-constraint devices, which can enable local processing at Edge devices. Another important issue related to distributed data analytics is modifying the existing code for different devices and application scenarios. The lack of standardization also creates interoperability issues in sharing data among heterogeneous devices, which makes the problem more challenging.

Edge Mesh has been proposed for IoT applications, and thus issues related to Internet of Things also contribute towards major research challenges in implementing Edge Mesh. IoT related issues have been surveyed in a lot of papers such [8], [11], [26], and [47]. Some of the major research issues related to IoT include architecture design, addressing and mapping, device management, device mobility, routing, data management, scalability, interoperability, security and privacy, etc. The issue that is most prominent with IoT is that devices are heterogeneous, resource-constraint, and the scale is huge. The scalability is one major challenge associated with IoT which makes other research issues more complicated. Edge Mesh should support management of large numbers of remote devices, which is quite challenging as it includes many actions including controlling the devices, configuring the devices, monitoring its status such as connectivity information, gathering data, etc. Security and privacy is another issue that becomes more complicated in Edge Mesh due to its distributed approach [48]. There are many security and privacy issues including identity and authentication, access control, protocol and network security, trust and governance, etc. [48]. Distributed algorithms are required for Edge Mesh to secure against attacks such as denial of service, eavesdropping, etc.

VII. CONCLUSION AND FUTURE WORK

This paper proposes a new computing paradigm, Edge Mesh, which focuses on enabling distributed intelligence in IoT. Edge Mesh distributes the whole application into sub-tasks which are distributed among Edge devices. Edge devices together with routers form a mesh network which is responsible for many computation tasks such as storage, processing, data sharing, etc. Edge Mesh tries to integrate best features from Cloud computing, Fog computing, and cooperative computing to provide multi-dimensional features. This paper also proposes a software framework for Edge Mesh. Software framework is divided into three levels corresponding to End devices, Edge Mesh, and Cloud. A task management framework for distributing and managing has also been discussed in detail.

We have pointed out various open challenges in developing Edge Mesh computing paradigm. Traditional analytics algorithm developed for centralized systems or distributed systems cannot be directly used for Edge Mesh as the communication channel is dynamic and devices used in Edge Mesh are very different from those traditional systems. Edge devices used for computation tasks are resource-constraint and heterogeneous. Besides, Edge Mesh also requires interaction and data exchange between different types of devices such as end devices, routers, edge devices, and cloud, which is another major challenge. We have discussed various benefits provided by Edge Mesh such as low latency, distributed processing, fault tolerance, better security and privacy, better services due to context awareness, etc. We have also discussed different application scenarios in three different application domains, Smart Home, Intelligent transportation systems, and Healthcare, to illustrate the significance of Edge Mesh.

We have also done a preliminary study of task allocation problem in Edge Mesh which aims to distribute dependent tasks in a complex application onto the Edge devices such that total energy consumption is minimized. The problem considered in this work is different from existing works related to task allocation as we have considered data distribution for dependent tasks. So far, we have only proposed a genetic algorithm to solve the task allocation and extensive experimentation is required to do thorough evaluation. New heuristic algorithms need to be proposed and performance comparison should be done with other approaches. We also need to consider other constraints and objective functions for this problem. We will consider these issues with the task allocation problem in our future works. We will also work on the implementation of the software framework for Edge Mesh and test its efficacy using different application scenarios discussed in the paper. Edge Mesh opens various research opportunities for researchers such as developing distributed algorithms that can be executed on resource constraint Edge devices. Task allocation problem discussed in this paper is just one example of new research challenges associated with Edge Mesh. Many research challenges discussed in this paper such as enabling meaningful exchange of data between heterogeneous devices, proposing distributed security and privacy algorithms for resource constraint devices, enabling integration of different computing paradigms, etc. are all open issues that require more research efforts.

REFERENCES

[1] Y. Qin, Q. Z. Sheng, N. J. G. Falkner, S. Dustdar, H. Wang, and A. V. Vasilakos, "When things matter: A survey on data-centric Internet of Things," *J. Netw. Comput. Appl.*, vol. 64, pp. 137–153, Apr. 2016.

[2] C.-W. Tsai, C.-F. Lai, C.-F. Lai, and L. T. Yang, "Data mining for Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 77–97, 1st Quart., 2014.

[3] D. Gil, A. Ferrández, H. Mora-Mora, and J. Peral, "Internet of Things: A review of surveys based on context aware intelligent services," *Sensors*, vol. 16, no. 7, p. 1069, 2016.

[4] Q. Wu *et al.*, "Cognitive Internet of Things: A new paradigm beyond connection," *IEEE Internet Things J.*, vol. 1, no. 2, pp. 129–143, Apr. 2014.

[5] A. I. Maarala, X. Su, and J. Riekki, "Semantic reasoning for context-aware Internet of Things applications," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 461–473, Apr. 2016.

[6] F. Van den Abeele, J. Hoebeke, G. K. Teklemariam, I. Moerman, and P. Demeester, "Sensor function virtualization to support distributed intelligence in the Internet of Things," *Wireless Pers. Commun.*, vol. 81, no. 4, pp. 1415–1436, 2015.

[7] M. Stolpe, "The Internet of Things: Opportunities and challenges for distributed data analysis," *ACM SIGKDD Explorations Newslett.*, vol. 18, no. 1, pp. 15–34, 2016.

[8] R. Alur *et al.* (2016). "Systems computing challenges in the Internet of Things." [Online]. Available: https://arxiv.org/abs/1604.02980

[9] I. Stojmenovic, "Machine-to-machine communications with in-network data aggregation, processing, and actuation for large-scale cyber-physical systems," *IEEE Internet Things J.*, vol. 1, no. 2, pp. 122–128, Apr. 2014.

[10] A. Saeed, M. Ammar, K. A. Harras, and E. Zegura, "Vision: The case for symbiosis in the Internet of Things," in *Proc. 6th Int. Workshop Mobile Cloud Comput. Services*, 2015, pp. 23–27.

[11] S. Ray, Y. Jin, and A. Raychowdhury, "The changing computing paradigm with Internet of Things: A tutorial introduction," *IEEE Des. Test*, vol. 33, no. 2, pp. 76–96, Apr. 2016.

[12] P. Mell *et al.*, "The NIST definition of cloud computing," Nat. Inst. Standards Technol., U.S. Dept. Commerce, Gaithersburg, MD, USA, Tech. Rep. 800-145, 2011.

[13] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and Internet of Things: A survey," *Future Generat. Comput. Syst.*, vol. 56, pp. 684–700, 2016.

[14] OpenFog. *Definition of Fog Computing*. Accessed on May 14, 2017. [Online]. Available: http://www.openfogconsortium.org/resources/#definition-of-fog-computing

[15] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.

[16] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for Internet of Things and analytics," in *Big Data Internet Things: A Roadmap for Smart Environments*. Cham, Switzerland: Springer, 2014, pp. 169–186.

[17] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[18] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. (2017). "Mobile edge computing: Survey and research outlook." [Online]. Available: https://arxiv.org/abs/1701.01090

[19] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. Workshop Mobile Big Data*, 2015, pp. 37–42.

[20] M. Villari, M. Fazio, S. Dustdar, O. Rana, and R. Ranjan, "Osmotic computing: A new paradigm for edge/cloud integration," *IEEE Cloud Comput.*, vol. 3, no. 6, pp. 76–83, Nov./Dec. 2016.

[21] A. Munir, P. Kansakar, and S. U. Khan. (2017). "IFCIoT: Integrated fog cloud IoT architectural paradigm for future Internet of Things." [Online]. Available: https://arxiv.org/abs/1701.08474

[22] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun. (2015). "Fog computing: Focusing on mobile users at the edge." [Online]. Available: https://arxiv.org/abs/1502.01815

[23] I. Farris, L. Militano, M. Nitti, L. Atzori, and A. Iera, "MIFaaS: A mobile-IoT-federation-as-a-service model for dynamic cooperation of IoT cloud providers," *Future Generat. Comput. Syst.*, vol. 70, pp. 126–137, May 2017.

[24] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.

[25] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Comput. Netw.*, vol. 47, no. 4, pp. 445–487, 2005.

[26] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," *Comput. Commun.*, vol. 54, pp. 1–31, Dec. 2014.

[27] Cisco. (Apr. 2015). *Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are*. Accessed on May 14, 2017. [Online]. Available: http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf

[28] R. Yang and M. W. Newman, "Learning from a learning thermostat: Lessons for intelligent systems for the home," in *Proc. 2013 ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2013, pp. 93–102.

[29] Y. Jiang, "A survey of task allocation and load balancing in distributed systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 2, pp. 585–599, Feb. 2016.

[30] J. Yang, H. Zhang, Y. Ling, C. Pan, and W. Sun, "Task allocation for wireless sensor network using modified binary particle swarm optimization," *IEEE Sensors J.*, vol. 14, no. 3, pp. 882–892, Mar. 2014.

[31] B. Billet and V. Issarny, "From task graphs to concrete actions: A new task mapping algorithm for the future Internet of Things," in *Proc. IEEE 11th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Oct. 2014, pp. 470–478.

[32] K. Ranganathan and I. Foster, "Decoupling computation and data scheduling in distributed data-intensive applications," in *Proc. 11th IEEE Int. Symp. High Perform. Distrib. Comput. (HPDC)*, Jul. 2002, pp. 352–358.

[33] J. Taheri, A. Y. Zomaya, H. J. Siegel, and Z. Tari, "Pareto frontier for job execution and data transfer time in hybrid clouds," *Future Generat. Comput. Syst.*, vol. 37, pp. 321–334, Jul. 2014.

[34] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, and A. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 4, pp. 23–32, 2013.

[35] S. Feng, P. Setoodeh, and S. Haykin, "Smart home: Cognitive interactive people-centric Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 34–39, Feb. 2017.

[36] S. Makonin, L. Bartram, and F. Popowich, "A smarter smart home: Case studies of ambient intelligence," *IEEE Pervasive Comput.*, vol. 12, no. 1, pp. 58–66, Jan. 2013.

[37] S. Gokceli, N. Zhmurov, G. K. Kurt, and B. Ors, "IoT in action: Design and implementation of a building evacuation service," *J. Comput. Netw. Commun.*, vol. 2017, Jan. 2017, Art. no. 8595404.

[38] H. M. Poy and B. Duffy, "A cloud-enabled building and fire emergency evacuation application," *IEEE Cloud Comput.*, vol. 1, no. 4, pp. 40–49, Nov. 2014.

[39] M. A. A. Pedrasa, T. D. Spooner, and I. F. MacGill, "Coordinated scheduling of residential distributed energy resources to optimize smart home energy services," *IEEE Trans. Smart Grid*, vol. 1, no. 2, pp. 134–143, Sep. 2010.

[40] J. S. Pan, I. S. Popa, and C. Borcea, "Divert: A distributed vehicular traffic re-routing system for congestion avoidance," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 58–72, Jan. 2017.

[41] K. Nellore and G. P. Hancke, "A survey on urban traffic management system using wireless sensor networks," *Sensors*, vol. 16, no. 2, p. 157, 2016.

[42] A. M. de Souza and L. A. Villas, "A fully-distributed traffic management system to improve the overall traffic efficiency," in *Proc. 19th ACM Int. Conf. Modelling, Anal. Simulation Wireless Mobile Syst.*, 2016, pp. 19–26.

[43] J. Pan, M. A. Khan, I. S. Popa, K. Zeitouni, and C. Borcea, "Proactive vehicle re-routing strategies for congestion avoidance," in *Proc. IEEE 8th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, May 2012, pp. 265–272.

[44] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

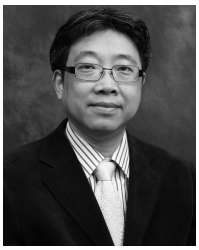
[45] D. Bradford and Q. Zhang, "How to save a life: Could real-time sensor data have saved Mrs Elle?" in *Proc. CHI Conf. Extended Abstracts Hum. Factors Comput. Syst.*, 2016, pp. 910–920.

[46] D. Datla *et al.*, "Wireless distributed computing: A survey of research challenges," *IEEE Commun. Mag.*, vol. 50, no. 1, pp. 144–152, Jan. 2012.

[47] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of Things: Vision, applications and research challenges," *Ad Hoc Netw.*, vol. 10, no. 7, pp. 1497–1516, Sep. 2012.

[48] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed Internet of Things," *Comput. Netw.*, vol. 57, no. 10, pp. 2266–2279, 2013.

YUVRAJ SAHNI received the B.E. degree (Hons.) in electrical and electronics engineering from the Birla Institute of Technology and Science, Pilani, India, in 2015. He is currently pursuing the Ph.D. degree with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. His research interests include wireless sensor networks, middleware, and Internet of Things.



JIANNONG CAO (F'15) received the B.Sc. degree in computer science from Nanjing University, China, in 1982, and the M.Sc. and Ph.D. degrees in computer science from Washington State University, USA, in 1986 and 1990 respectively. He is currently a Chair Professor with the Department of Computing at The Hong Kong Polytechnic University, Hong Kong. He has co-authored five books in mobile computing and wireless sensor networks, co-edited nine books, and published over 500 papers in major international journals and conference proceedings. His research interests include parallel and distributed computing, wireless networks, and mobile computing, big data and cloud computing, pervasive computing, and fault tolerant computing. He is a member of the ACM and a Senior Member of the China Computer Federation.

SHIGENG ZHANG (M'11) received the B.Sc., M.Sc., and D.Eng. degrees in computer science from Nanjing University, China, in 2004, 2007, and 2010, respectively. He is currently an Associate Professor with the School of Information Science and Engineering, Central South University, China. He has authored over 50 technique papers in top international journals and conferences, including Infocom, ICNP, TMC, TC, TPDS, TOSN, and JSAC. His research interests include Internet of Things, mobile computing, RFID systems, and Internet of Things security. He is on the Editorial Board of the International Journal of Distributed Sensor Networks. He was a program committee member of many international conferences, including ICPADS, MASS, and ISPA. He is a member of the ACM.

LEI YANG received the B.Sc. degree from Wuhan University, in 2007, the M.Sc. degree from the Institute of Computing Technology, Chinese Academy of Sciences, in 2010, and the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University, in 2014. He is currently an Associate Professor with the School of Software Engineering, South China University of Technology. His research interest includes computer networks, mobile cloud computing, and big data analytics.