IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 28, 2017, accepted October 16, 2017, date of publication October 26, 2017,
date of current version December 5, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2766923

A Framework of Fog Computing: Architecture, Challenges, and Optimization

YANG LIU, JONATHAN E. FIELDSEND, (Member, IEEE), AND GEYONG MIN, (Member, IEEE)
Department of Computer Science, University of Exeter, Exeter EX4 4QF, U.K.

Corresponding author: Geyong Min (g.min@exeter.ac.uk)

This work was supported in part by the Engineering and Physical Sciences Research Council under Grant EP/P020224/1 and in part by the EU FP7 QUICK Project under Grant PIRSES-GA-2013-612652. The work of Y. Liu was supported by the Chinese Scholarship Council.

ABSTRACT Fog computing (FC) is an emerging distributed computing platform aimed at bringing computation close to its data sources, which can reduce the latency and cost of delivering data to a remote cloud. This feature and related advantages are desirable for many Internet-of-Things applications, especially latency sensitive and mission intensive services. With comparisons to other computing technologies, the definition and architecture of FC are presented in this paper. The framework of resource allocation for latency reduction combined with reliability, fault tolerance, privacy, and underlying optimization problems are also discussed. We then investigate an application scenario and conduct resource optimization by formulating the optimization problem and solving it via a genetic algorithm. The resulting analysis generates some important insights on the scalability of the FC systems.

INDEX TERMS Fog computing, genetic algorithms, Internet of Things, optimization.

I. INTRODUCTION

Along with the developments of the Internet-of-Things (IoT) itself and relevant techniques, the concept of IoT is evolving. Most of the literature identifies Kevin Ashton, a British technology pioneer, as coining the phrase ''Internet of Things'' as the title of a presentation he made at Procter & Gamble in 1999. Since then the term ''IoT'' has been popularized and is now widely used. The US, EU, China, Japan, and Korea have all proposed national level projects to develop IoT. As more application domains have been explored, researchers find that more and more technologies are encompassed by IoT. Therefore, the concept from ''things to things'' expands to ''things to things, things to humans, humans to humans'' or even ''everything to everything''. For serving mass applications, the architectures of IoT are well studied. From the very beginning, the architecture is a three layer design, e.g. IEEE P2413, EU CSAGRAS and CCSA's architecture. Later on, many derived architectures with five layers [1], [2] were proposed. In this paper, a general four layer architecture with big data awareness is discussed, as illustrated in Fig. 1. The most important reason for treating data processing independently from applications is that the cloud computing service emerges often as a third party. In truth, massive data processing is still a bottleneck of IoT. For years, some researchers have expected cloud computing to solve this problem and proposed many cloud based IoT architectures [3]–[6].

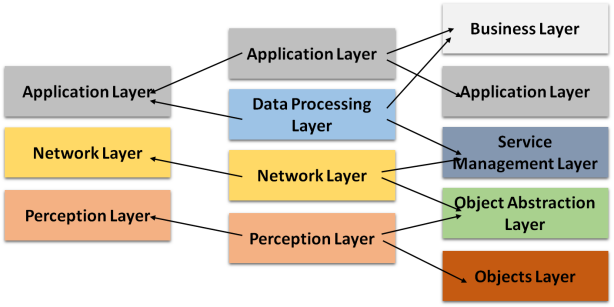

FIGURE 1. Four layer architecture and the mapping with three and five layers.

In the survey of [7], the application is mainly divided into three categories: industry, environment and society. Libelium even split the category into further detailed levels [8]. 61 applications are clustered in 12 categories. This embodies the perspective from industry. Mass applications include RFID and sensor networks. Currently, although IoT data are already inferred as big data, video data such as surveillance video are considered as the 'largest' big data [9], which can easily make the IoT data grow to TB/PB level in seconds. Almost every application with video may meet challenges

 This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/

in computing, for instance, encoding/decoding, recognition, motion capture, video completion, etc.

The phenomenon of big data is rapidly promoting the IoT business to an advanced level: having to cope with both large scale and high complexity. These technical considerations also bring some concerns about the latency sensitivity, which restricts the expansion of IoT to some extent [12], [13]. Bonomi *et al.* [14] discussed this issue in the relationship between time scales, from milliseconds to months, and service levels, from sensors to business data repositories.

The reasons for little latency sensitivity of the traditional cloud computing architecture are embodied in two perspectives: networking and data. From the networking aspect, the traffic converges toward the data center from each subsystem of a large scale IoT system, e.g. smart city, with a pyramid like systematic organization. The networks near the center can therefore suffer traffic overload due to the data deluge, in turn causing a series of network problems including critical delay. Even when counting the time cost of long distance transmission, cloud based architecture *cannot* guarantee the latency sensitive requirements for many IoT applications. Additionally, data itself is an encumbrance for both networking and computing. In some models of IoT, preprocessing systems are proposed to slim the data. This is typically accomplished via removing the redundancy that inevitably contains a great deal of overt and latent semantics and contexts generated from the data sources [15]. However, in the perspective of big data domain, redundancy can play an important role in exploring some underlying/unexpected information and discarded such data may lead to a degradation in prediction accuracy.

Although latency sensitivity and big data appear incompatible, we *cannot* separate these two terms because the era of latency sensitive big data is coming. In popular projects on self-driving cars, e.g. Google's robotic cars, the data generated from complicated surroundings via various sensors and cameras is massive (approximately over 1Gb/s). The processors must give accurate orders to the steering system in milliseconds by computing these data. Until the technique is mature, an onboard computer alone is insufficient and cloud participation becomes indispensable. However, if the data is delayed in a cloud server due to queueing or networking failure, a ''smart'' car may lose its intelligence and cause accidents.

In order to release the dual burdens from networking and data, and then achieve intelligent goals, some organizations have given their prophesies or solutions. IDC Future Scape 2015 [16] has predicted that ''IoT at the edge'' which refers to marginalization of data processing step in IoT would occupy a large proportion, ''40%'' in the source report. Edge computing technologies that can push computing and networking services away from centralized facilities would appear to accordingly be a point of focus. In what follows, some related platforms are introduced.

1) UBIQUITOUS COMPUTING

Ubiquitous Computing (UC) [17] is a classic computing concept where the basic idea is to use terminals including mobile phone, sensor, actuator, wearable equipment, gateway, and access point to operate lightweight data processing. It is widely accepted that UC is the foundation of IoT. According to e.g. [18], UC has high level mobility and embeddedness. Generally, UC's appearance is as the overlap between mobile computing and edge computing.

2) CLOUDLET

Satyanarayanan *et al.* [13] introduced the concept of cloudlets: ''trusted, resource rich computers in the near vicinity of mobile users''. The goal of cloudlet is explicit, i.e. ''bringing the cloud closer''. Verbelen *et al.* [19] analyzed the two drawbacks of the virtual machine (VM) based cloudlet approach and introduced two architectures with corresponding solutions: adhoc cloudlet and elastic cloudlet. The role of these two is mainly for computation offloading. Cloudlet can be regarded as the overlap between cloud computing and edge computing.

3) MOBILE CLOUD COMPUTING

Due to the popularity of computation offloading in a mobile environment, Mobile Cloud Computing (MCC) [20] has become a hot topic recently. MCC processes a part of the task locally and migrates the remainder to a high performance cloud center with the main perspective of energy efficiency. In essence, MCC is close to the crossover between cloud computing and mobile computing.

4) FOG COMPUTING

Fog Computing (FC), firstly introduced by Cisco [21], emerges as a novel topic expected to solve the latency sensitive computing problems. Similar to some other edge computing platform, FC utilizes local computing resources instead of a remote cloud for data processing. The geographic proximity between the data source and processors reducing the transmission latency. Some research efforts have been made in [22] and [24] and discussed some ideas regarding FC and its motivational role in IoT. The illustration of FC mapping to IoT is shown in Fig. 2(a). The function of FC is to leverage the local computing resources to process tasks.

Luan *et al.* [22] compared FC and cloud computing (including cloudlet) and highlighted three features: wireless, local service, and distributed management. However, these features also belong to UC and *cannot* distinguish the new concept from the classical one. In our view, another key feature that can represent the distinction of FC, also revealed by the aforementioned auto driving project, should be claimed: it is complex application oriented. In reality, many complicated computing tasks are far beyond the ability of a single hardware in UC since it may cost an unbearable runtime. While, if FC is exploited, a group of cheap and low performance single

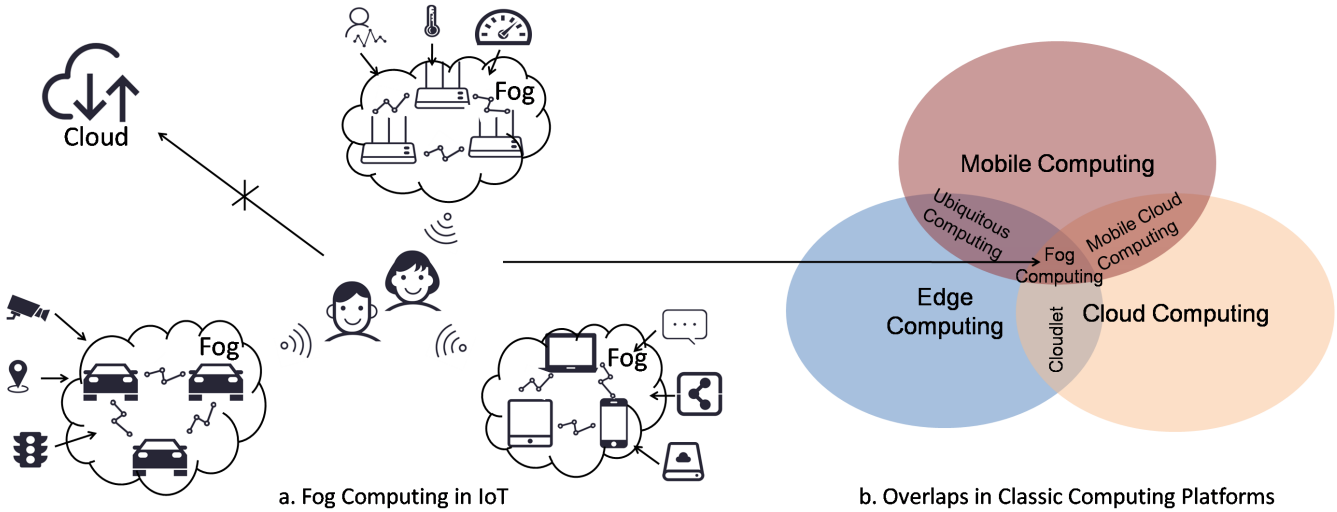

FIGURE 2. Description of fog computing: Fog computing is the central overlap of mobile computing, edge computing, and cloud computing.

TABLE 1. A comparison of service platforms.

Platform	Mobility	Freedom	Cost	Computation ability	Latency
Cloudlet	Medium	N/A	Medium	High	Medium
Mobile cloud	High	N/A	High	Very High	High
Ubiquitous	High	High	Low	Low	Low
Fog	High	Medium	Low	Medium	Low

hardwares can constitute a high performance platform, which is similar to why cloud computing was conceived.

Also, from geo-distributed view, the "fog" should be positioned at the gateway level. The advantages of using the gateway are discussed as follows. Firstly, its computing power could be better than the terminal can provide. In thousands of IoT applications, the gateway is developed with Advanced RISC Machines (ARM) that have a powerful capacity, even close to PC level. And the terminal device often uses a single chip, e.g. C51, with only an 8 bit microcontroller. Secondly, the gateway often uses consistent electricity for its vital role, which can negate the energy consumption concern when optimizing the performance of computing and networking. Thirdly, unlike a mobile phone, gateways are sometimes deployed for public services and self managed, which means the selflessness and privacy concerns may be partly reduced by a black box mechanism.

Consequently, in this article we make a definition of FC, aiming to distinguish this emerging platform from the existing ones: *Fog Computing is a wireless distributed computing platform in which complex latency sensitivity tasks can be processed via a group of sharing resources at IoT gateway level in a locality.* As shown in Fig. 2(b), we prefer the central overlap of mobile computing, cloud computing and edge computing as the description of FC rather than embracing them all into a single concept.

Tab. 1 shows the comparisons between general cloudlet, MCC, UC, and FC from various aspects. From this we can easily find out the advantages and disadvantages of these service modes.

In addition to providing the definition, the framework of FC will now be proposed with an architecture, modeling, and solutions. The main contributions of this article include:

- The architecture of FC comprising both computing and networking aspects is provided to bring the distinctions of FC into full play.
- We discuss the potential issues regarding the latency sensitive problem, with the consideration of the characterisation and requirements of an FC service and summarize the underlying solutions about them.
- We use an application scenario to describe how FC works and how to realize the latency reduction with modeling and optimization of resource allocation and subtask scheduling.

II. ARCHITECTURE OF FOG COMPUTING

The general mode of FC is shown in Fig. 3. The procedure of FC service is described as follows:

1) The data is first partitioned into chunks.
2) The chunks are allocated to participating nodes.
3) The chunks are queued before transmission.
4) Based on the queue, the channels are allocated, which makes some chunks occupy the idle channels in the first batch and rest of them wait for the next released channel.
5) After the distributed processing, the processed chunks are sorted by their finishing time.
6) Also using channel allocation, these chunks are returned to the host.
7) Finally, the chunks are reunited by the host.

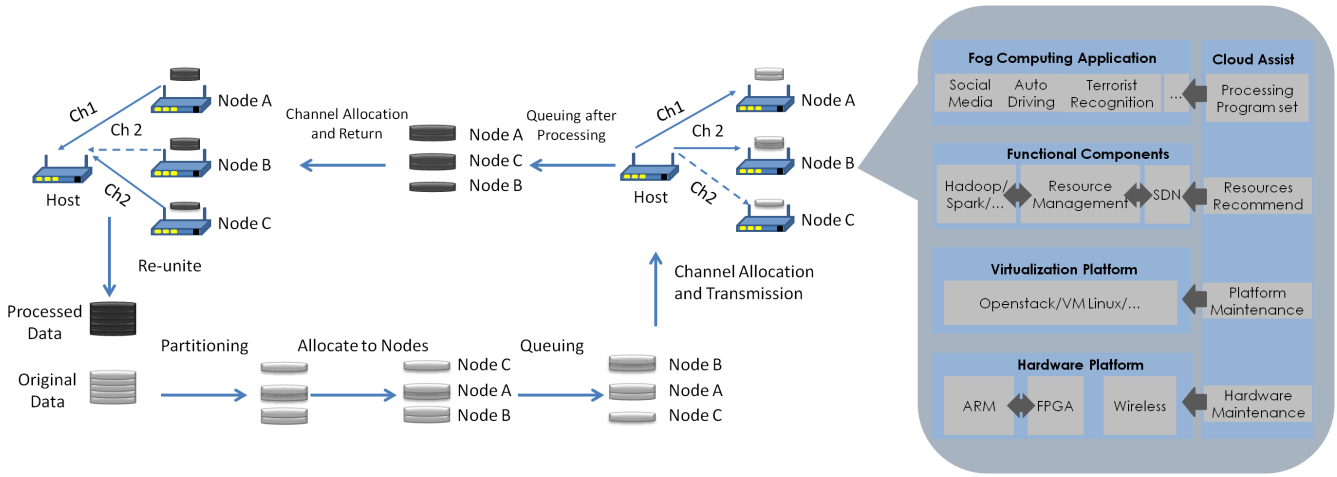

FIGURE 3. Architecture of fog computing: The procedures from original data to processed data and the design of fog node.

The fog node design is also based on the architecture of FC. According to the nature of FC, the architecture will be elaborated in two parts: computing and networking.

A. COMPUTING ARCHITECTURE

1) HARDWARE PLATFORM

Although some related hardwares are already designed, in order to leverage the power of FC, the hardware platform still needs discussion. Two premises emerge from the nature of FC. Firstly, since the platform is distributed, easy-to-manage becomes a prioritized consideration. In fact, easy-to-manage is a broad request that includes rich interfaces, easy programming, abundant softwares, etc. To satisfy these requests, ARM usually becomes the first choice for an embedded system. The ARM family has already been well applied in thousands of applications in IoT. Secondly, with an ultimate goal of computing, the processing speed is a key point. Although ARM cores now have excellent processing power, doubts about the performance still exist when facing complex computing. However, a new HPC computing resource has recently been funded in the UK to provide computing resource for UK scientists using 64-bit ARM CPUs [23], so the views on the capability of ARM cores in this arena may well change. Field Programmable Gate Array (FPGA), another well known hardware alignment in IoT, can naturally come to mind since it can process complex algorithms fast in a parallel way, which perfectly matches the idea for accelerating big data applications. Even further, Xilinx known for inventing FPGA has launched Zynq series production that integrates ARM and FPGA together; ARM takes the charge of management and FPGA focuses on the algorithms. This solution is likely to be of great advantage to be a reference for setting up a fog node.

2) SOFTWARE PLATFORM

Bonomi *et al.* [14] discussed the components of fog software architecture and proposed the structures of abstraction and orchestration layer. Since Hadoop, Spark and Openstack are all announced to run on ARM by AMD, DataCentred, and other institutes or individuals, transplanting big data/cloud computing platform from PC/server to embedded hardware has become a feasible and agile approach. The abstraction layer maps the Openstack that virtualizes the heterogeneous resources with cloud structure. The orchestration layer is able to probe, analyze, plan and execute a job that can be implemented in a big data platform like Hadoop or Spark. For instance, Openstack is deployed upon the ARM core and uses Hadoop, Spark or more specific engine to work on it via APIs. The API design must consider flexibility, latency sensitivity and heterogeneity.

3) CLOUD ASSIST

Notwithstanding distributed deployment, FC needs a cloud assist mechanism for convenient management and maintenance which obviously are challenging to address manually on a local device. For example, the programs for data processing are initially installed in the RAM or ROM. If the programs or configurations need modification, the patch or advanced release will be downloaded from the cloud server. Besides this, the cloud is also helpful in the resource management. As a voluntary system, the participants are expected to be high performance, reliable and unselfish. Accordingly, the evaluation of these factors of a participant is very important and related to the historical behavior which naturally belongs to the duty of the cloud. When a user initiates a computing task, the cloud can push up a local list with the names of recommended resources.

B. NETWORKING ARCHITECTURE

1) WIRELESS TECHNOLOGY

Several wireless technologies are commonly used in IoT with a wide variety of performances. 3G/4G, an expensive broadband wireless technique, can provide a wide signal range but sometimes is not a necessity in the local

collaborative scenario of FC. In contrast, Zigbee, 6LoWPAN, wirelessHart, etc., well known protocols in wireless sensor networks (WSN), have low bandwidth, are short distance, with cheap hardware cost, and are free charged in spectrum. However, in order to enlarge the benefit of distributed systems under a wireless environment, the transmission speed is extremely important, otherwise the latency in transmission step may not be covered by the saving from computation. Although a number of heterogeneous wireless networks have been presented to balance between 3G/4G and WSN, homogeneous networks are generally recognized as the efficient solution in a consolidated platform. Amid all these wireless technologies, WLAN is a compromise choice for the preference of homogeneous networking architecture because of its balanced speed, cost and coverage. Nowadays, WLAN equipment can support the 802.11ac protocol with wave2 version, reaching 1 Gb/s. In addition, wave2 version can offer Multi user Multiple Input Multiple Output (MU-MIMO) mode to enhance the utility of wireless resources. MU-MIMO ideally satisfies the mode of distributing tasks towards multiple destination and implementing parallelized data processing with the concern of channel interference released. This technology also protects the security or privacy of data since the multiple antenna can provide beamforming to reduce the risk of signal leakage. Therefore, WLAN with MU-MIMO could be the base of a wireless module.

2) SINGLE HOP/ADHOC

For the networking connectivity, there are two different thoughts. In [22], the interior network is proposed based on single hop. However, Vaquero and Rodero-Merino [24] regard that fog nodes must be able to act as a router for its neighbors and be resilient to node churn and mobility, which was mostly referred to as adhoc mode. In essence, single hop is regarded to provide less latency in transmission; adhoc helps finding more appropriate computing resources so as to reduce the entire latency. If latency is the only concern, we can easily determine which mode is better via calculation with the known parameters. However, reliability as another nonnegligible factor also impacts the decision. For example, more links that are brought from the adhoc mode can upgrade the performance of the system, but can also increase the risk of losing connections. Even if one chunk fails, the entire task would be impeded. The reliability is a stochastic problem and should be modeled for the topological structure.

3) SDN

Software defined networking (SDN) is a highly focused topic in recent years. It can significantly reduce the required network administration and both the capital expense as well as operational expense and also provides fast service orchestration for the highly programmable framework from control plane to data plane. Many consulting reports highlighted SDN as a technology to be widely exploited in the future. Google's B4 project is regarded as one of the most successful implementations, which uses a combination of Quagga along with OpenFlow to optimize Google's own data center interconnects. Because the infrastructure of FC sometimes can be also seen as a wireless, embedded, and distributed data center, SDN may play an important role as discussed in the literature [22], [24].

As mentioned in Sec.II-A.2, the computing resources can implement virtualization using Openstack. Based on that, several SDN modules, e.g. OpenStack Neutron, OpenDaylight, etc., have also been released. Thus, SDN can be strongly supported by Openstack without any other stand alone modules integrated. Besides, centralized control via SDN can make up for the disadvantage of the unconsolidated nature of a distributed system. The initiator of a task that has its own network requirements can easily configure the parameters for itself and cooperative fog nodes using SDN since the constrains of the networking requirements from applications are various, e.g. robustness pursuit, privacy pursuit, etc.

III. LATENCY SENSITIVE PROBLEM

The orientation of FC is to enhance the latency sensitive performance of data processing. In spite of geographical features removing the long transmissions, task scheduling and resource allocation can further reduce the latency via modeling and optimization. However, they may be more complicated in FC environment than those in cloud computing or MCC.

A. LATENCY REDUCTION

Latency is always the main consideration in FC framework. If latency is well modeled and calculated in an optimum way, the latency sensitive performance can be enhanced. The latency of each data chunk can be divided into three components in general applications: distributing latency, processing latency, and return latency. The entire latency of the system, also presented by makespan depending on the back time of last chunk, can be written as

$$T_{makespan} = \max_i \left(\frac{c_i}{t_i} + \frac{c_i}{p_i} + \frac{c_i}{r_i} + g(c_i, \mathbf{p}, \mathbf{q}) \right) \quad (1)$$

where c_i is the size of the ith chunk of data, assumed to be processed on the ith processor, which can process p_i data units per unit of time. t_i is the amount of data that can be transmitted to the ith processor in a time unit, and r_i is the amount of data that can be returned by the processor per time unit (r_i can also include a factor to compensate for the returned amount differing from the originally received amount). $g(c_i, \mathbf{p}, \mathbf{q})$ is a nonlinear component that accounts for the waiting time experienced depending on the order $\mathbf{p}$ in the forward channels and $\mathbf{q}$ presents the return queue. In essence, the strategy of the entire procedure is to make the chunks tightly piled up. Hence, we need to allocate nodes with heterogeneous rates, channels, queues, and elastically partition the data into chunks.

The solution of Eq. (1) is a NP hard/complete problem depending on the resource allocation and subtask scheduling which are both traditional issues in computing and

networking domains. Heuristic approaches are potential methods for addressing them.

B. FAULT TOLERANCE

As aforementioned in the discussion of single hop/adhoc networking architecture, reliability is an important factor in resource discovery. Many existing studies on wireless networks, distributed computing, cloud computing, and peer to peer computing, especially along with the terms of scheduling, are based on fault tolerance which mainly uses extra resources to cover some accidents/failures. However, the topic in FC is still a serious challenge because it is sometimes obstructed by two unfavorable factors: multiple procedures and resource constraints.

Multiple procedures mean that we need to consider the possibility of fault in every step, not only in processing but also in the transmission forward and backward. It makes the fault tolerance more complicated compared with computing only or transmission only tasks. Resource constraints also influence the design of fault tolerant scheme. The primary backup model is often mentioned in this topic, which replicates the fragments of a task and lets each primary subtask have a backup for robustness [25]. But this method would degrade the overall performance if no failure occurs because the backup resources are wasted, if the quantity of fog nodes is limited as in the scenario of deploying gateways in IoT. In order to avoid occupying extra physical nodes, virtual machine (VM) mechanisms for backup are well studied instead of using physical nodes. A premise of these mechanisms is that the primary VM of a chunk *cannot* be deployed in the same physical nodes with the backup VM of it in this case. This approach is efficient in cloud computing with powerful nodes. However, the low capacity of fog nodes may degrade the performance severely when loading two or more VMs. Although some overlapping mechanisms are explored to save some VM resources, these approaches still need a tradeoff between latency minimization and task protection.

C. SECURITY CONCERN

Security can be the concern for every computer issue. However, only a few researches specifically aim at FC. Dsouza *et al.* [26] focused on the sharing resource management and proposed a policy-driven security framework. Because security always needs a systematic solution, more security techniques, traditional or novel, are expected to suit FC. As a part of IoT, FC has some similar characteristics as mobile ad-hoc network (MANET), cloud computing, P2P and ubiquitous computing. The risk may integrate all the problems of these related technologies. The risk of MANET is summarized in [27]. In cloud computing, [28] present ''9 Worst Cloud Security Threats'': data breaches, data loss, account or service traffic hijacking, insecure APIs, denial of service, malicious insiders, abuse of cloud services, insufficient due diligence and shared technology. Although FC may not inherit all the threats, some of them surely have the potential to transfer. Stojmenovic and Wen [29] analyzed the man-in-the-middle attack in FC which is simple to launch but difficult to be addressed. Most of current solutions of such security problems in MANET domain, e.g. [30]–[33], are based on a cryptographic or trust scheme. But in the real world they are not always as secure as the original ideas due to e.g. implementation error.

a. Cryptographic mechanisms, which always play vital role for security, require a key management service to assist the establishment of mutual authentication between communication nodes. Also similar to wired network problems, the cryptographic methods could be cracked with high performance computing. Complex encryption and decryption techniques may not be suitable for some scenarios.
b. Trust is a lightweight security scheme which can transform complex security problems. It achieves this by providing the ability to ignore those (potential) participants who do not have trust within the system – preventing attack by preventing participation. Hence, if a security project exploits trust scheme, whether focusing on human participants or machines, risk is reduced. The management of trust can however be problematic. For trust evaluation scheme, how to model malicious behavior is still an open problem. Because a failed node caused by traffic congestion might be regarded as a malicious node and this kind of judgement can make the trust evaluation unreliable.
c. Some security risks exist which are not directly addressed by cryptographic or trust schemes. These include passive attacks that seriously affect user's privacy and active DoS (Denial of services) attacks, which focus on protocol vulnerability and network management.
d. The traditional broadcasting/multicasting method is omni-directional and attackers can obtain a chance to access useful information if they are located in the coverage area.

Physical mechanisms like smart antenna are an effective way to reduce the risk, as mentioned in the MU-MIMO discussion, and sandboxing mechanisms can protect the privacy against some cloud computing related threats. Despite this, security is a big challenge in the real application of FC.

D. PRIVACY CONCERN

Privacy is always a vital issue in distributed systems and big data [34], [35]. Besides some traditional techniques as reputation/trust based mechanism and cryptography, confining the size of chunks is a proactive way to reduce the risk of data leakage, which also impacts the sensitive latency issue. In some applications, the partitioning for the original task can break the global knowledge. But if a malicious node receives a large volume of chunk, it would get a great chance to infer the global knowledge. Restricted size of partitioning may lose the optimum of latency but should reduce the probability of any single node retrieving sensitive information.

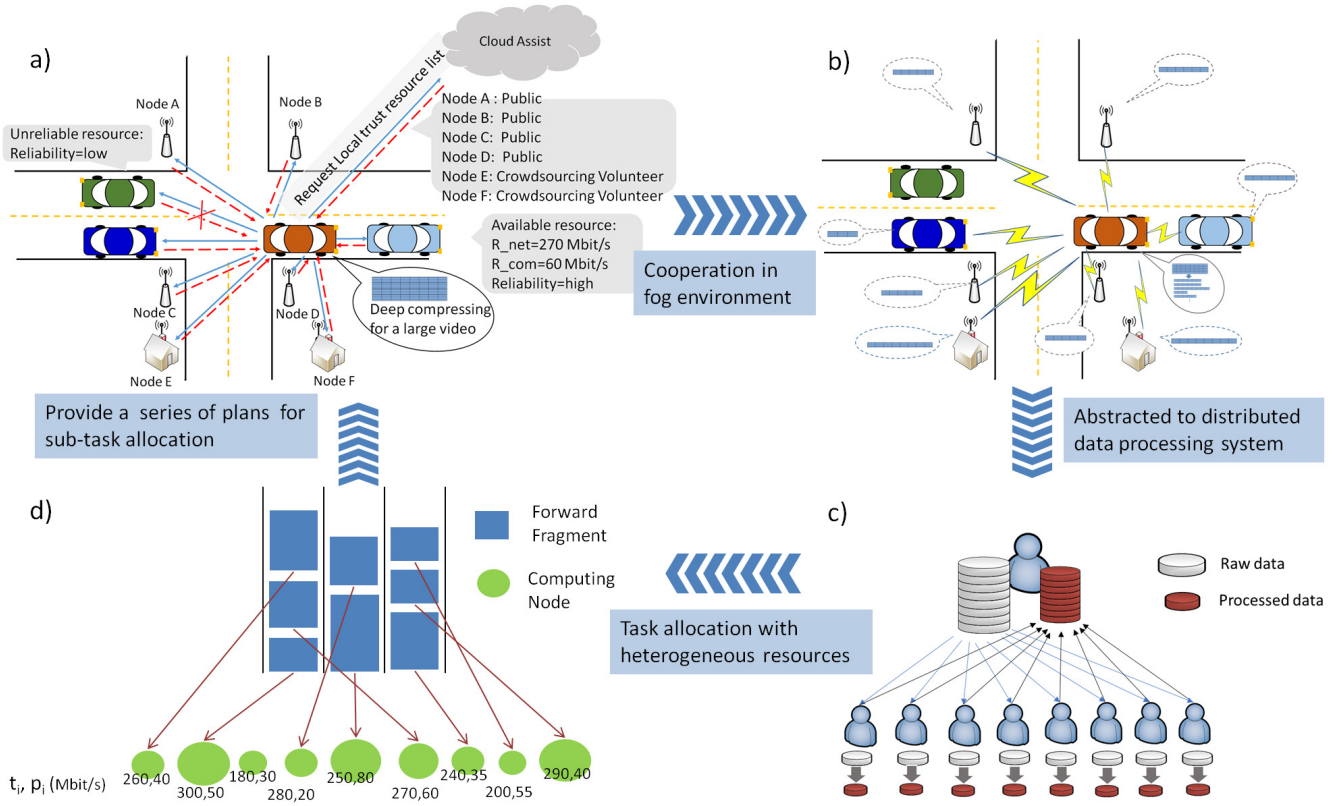

FIGURE 4. **Framework in application scenario: a) resources request; b) processing data with coalition; c) abstracted distributed processing; and d) task allocation via 3 MU-MIMO channels.**

As discussed, optimization of resource allocation and subtask scheduling can improve the performance with certain requirements. Optimization procedures are well studied in cloud computing areas, but whether the existing methods are adaptable to FC is not immediately clear since two elements, computing and networking resources, both have to be considered in the framework. Particularly, if MU-MIMO mode participates into the transmission step, the scheduling work would be much more complicated.

IV. EXAMPLE SCENARIO ON FOG COMPUTING

This section presents a scenario in the vehicular adhoc networks (VANET) application which usually catches research attentions for data delivery job only and here considers processing and transmission both in an FC environment. Under the proposed framework shown in Fig. 4, we will investigate the model of resource allocation, subtask scheduling and optimization for the purpose of latency reduction.

In the scenario, the user needs a deep compression for a large video from the car recorder in order to save the uploading cost via 4G networks. This work takes more than 30 minutes by using his/her own onboard computer, which is unacceptable for displaying in Youtube or Facebook in a timely manner. In this case, the eagerness for the involvement of FC is evoked. She/he initiates a coalition with the surrounding hardwares (fog nodes), i.e. public infrastructures and other vehicles. The requests of cooperation are broadcasted and the estimation of networking rate for responding nodes is made simultaneously. Meanwhile, with the assistance of the remote cloud, a list of local trust nodes is downloaded and is labeled as prime choices. As shown in Fig. 4(a), some trust nodes are government furnished and some are from volunteering individuals. While, the resources which are out of the list needs a further selection via the context aware information that provides an evaluation of the reliability of these nodes. For instance, if a node moves towards an opposite direction, which means it has high risk to drop the link during the task period, it would be recognized as a ''low'' reliable node and be excluded. The other high reliablity nodes report their computation ability in the feedback. A task initiator gathers the feedbacks and selects the feasible nodes in the coalition. Afterwards, the video is partitioned into fragments under a certain strategy and dispersed to the allocated cooperators, as Fig. 4(b). The strategy includes: How to partition the task into subtasks and schedule the subtasks in order to minimize the latency in the entire process. The procedures can be abstracted as distributed data processing as shown in Fig. 4(c). Three parts are contained: distributing, processing, and return. This work is apparently a resource allocation and task scheduling problem, illustrated in Fig. 4(d). MU-MIMO is adopted in the framework since it can provide separated

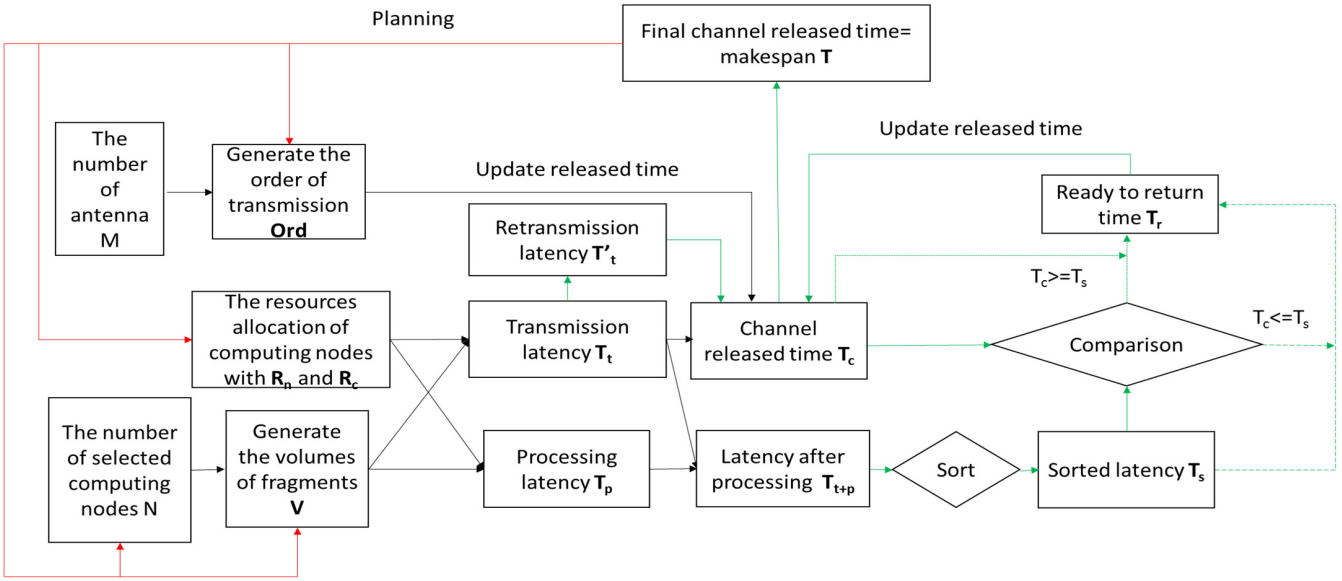

FIGURE 5. The process of latency calculating.

channels with full rate, which is of great advantage in wireless distributed environment. If the strategy is reasonable via modeling and optimization, the latency will be reduced. The optimization provides a series of plans about how the initial task is partitioned, where the chunks are dispatched, and how the chunks are queued. The process of latency calculation is shown in Fig. 5. The objective of the process is to identify an efficient plan of chunk size, order of delivery, nodes, and channels under the minimized latency.

A heuristic approach can be induced to optimize the problem based on the latency model as Eq. (1). For example, a Genetic Algorithm (GA) approach with two composites, elastic partitioning and an order of transmission, can be utilized. The order of transmission is a classic permutation problem in GA applications. While, the partitioning problem needs a little more work to insert into the GA framework. Actually, to implement elastic partitioning, we can use Dirichlet distribution to generate a set of random vectors: Let

$$X = (X_1, X_2, X_3, ..., X_K) \sim Dir(\alpha)$$

satisfy

$$\sum_{i=1}^{K} X_i = 1, X_i \in (0, 1)$$

where α is concentration parameter. In GA, we just substitute the mutation step of partitioning part with sampling from a Dirichlet distribution which generates a group of random 'sum to one' vectors.

In order to verify the idea of optimization under the framework. A numerical test with 10, 20, 30, 40 and 50 single hop nodes has been implemented. Also we use 3x3 MU-MIMO technique to offer three isolated high speed channels. The 50 groups of coupled heterogeneous rates are set

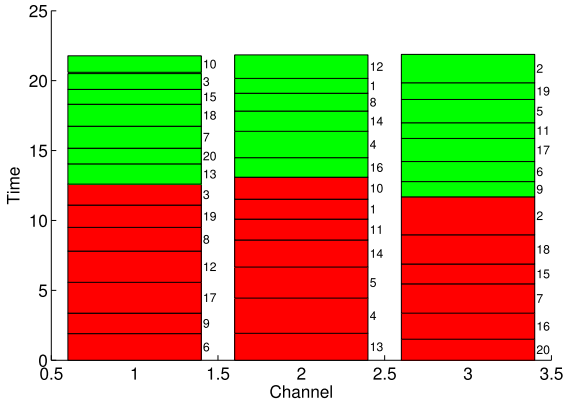

FIGURE 6. The schedule scheme. Red regions denote outbound transmission times and green regions inbound transmission times.

by random values which are generated with [200, 300] and [20, 100] ranges respectively for processing and transmitting so as to simulate a complex processing service as the scenario above. When utilizing fewer than all 50 nodes, e.g. for the 10 nodes example, the preselection of top 10 nodes that provide the better overall performance to be achieved by a simple function (e.g. multiplication) would be done first. The whole data volume is 10GB. From Fig. 5, we can find that the latencies are improved for all the cases, more than 1.5s for 20∼50 nodes and even 0.8s reduction for 10 nodes. The minimum latency can reach 21.5s for 30, 40 and 50 nodes cases.

The improvement trends are also shown. The case of 20 nodes provides much better performance than 10 nodes case. However, the improvement is not apparent when the number of nodes increases further. It is worth noting that the initial values for more nodes are worse than those with fewer

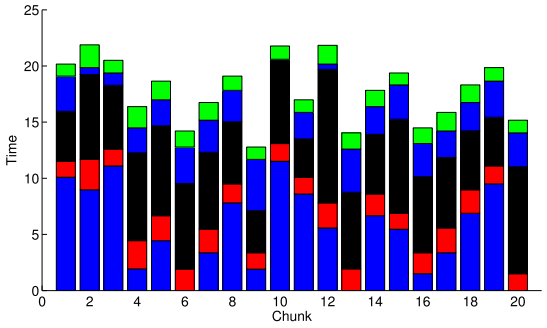

FIGURE 7. The channel usage. Red regions denote outbound transmission times, black regions processing times, green regions inbound transmission times, and blue regions denote blocking (waiting) times. The GA population size was 100 and it was run for 1000 generations.

nodes. The reason is the high performance nodes are selected among all the nodes in the cases of fewer nodes; when the number of nodes increases, some 'poor' nodes have to be used in the coalition and drag down the overall performance. While, along with the iterations of optimization, the gaps of initial makespan are reduced, which shows the power of optimizer. Even so, the scale of FC should be carefully considered with concerns on performance, reliability, and privacy in practice. Fig. 6 and 7 illustrate the schedule and channel usage in 20 nodes case. The chunk size, permutation, channel usage, and waiting time can be seen clearly.

The research materials supporting this publication can be accessed at the Open Research Exeter repository: https://ore.exeter.ac.uk/repository/.

V. CONCLUSION

Fog Computing (FC) is an emerging local distributed computing platform under a wireless and embedded environment. In this article, we present the characters of FC and discuss its differences from other similar computing platforms. Firstly, an architecture of FC in both computing and network aspects is presented. Secondly, a framework for resource allocation and latency reduction is proposed. Meanwhile, fault tolerance and privacy are both considered in the framework with the corresponding potential solutions or optimization methods. Finally, we evaluate the framework under a given application scenario and Genetic Algorithm combined with a Dirichlet distribution sampling approach.

REFERENCES

[1] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future Internet: The Internet of Things architecture, possible applications and key challenges," in *Proc. FIT*, Dec. 2012, pp. 257–260.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[3] W. Lumpkins, "The Internet of Things meets cloud computing [standards corner]," *IEEE Consum. Electron. Mag.*, vol. 2, no. 2, pp. 47–51, Apr. 2013.

[4] H.-L. Truong and S. Dustdar, "Principles for engineering IoT cloud systems," *IEEE Cloud Comput.*, vol. 2, no. 2, pp. 68–76, Mar./Apr. 2015.

[5] G. Hurlburt, I. Bojanova, and R. Berezdivin, "Computational networks: Challenging traditional program management," *IT Prof.*, vol. 16, no. 6, pp. 66–69, Nov. 2014.

[6] J. Jin, J. Gubbi, S. Marusic, and M. Palaniswami, "An information framework for creating a smart city through Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 2, pp. 112–121, Apr. 2014.

[7] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 414–454, 1st Quart., 2014.

[8] Libelium. (2015). *50 Sensor Applications for a Smarter World*. [Online]. Available: http://www.libelium.com/top 50 iot sensor applications ranking/

[9] T. Huang, "Surveillance video: The biggest big data," *Computing Now*, vol. 7, no. 2, pp. 82–91, Feb. 2014.

[10] A. Pentland, "Looking at people: Sensing for ubiquitous and wearable computing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 107–119, Jan. 2000.

[11] J. Ren, Y. Zhang, K. Zhang, and X. Shen, "Exploiting mobile crowdsourcing for pervasive cloud services: Challenges and solutions," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 98–105, Mar. 2015.

[12] Gartner. (2014). *Gartner Says the Internet of Things Will Transform the Data Center*. [Online]. Available: http://www.gartner.com/newsroom/id/2684616

[13] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct./Dec. 2009.

[14] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, *Fog Computing: A Platform for Internet of Things and Analytics* (Studies in Computational Intelligence in Big Data and Internet of Things: A Roadmap for Smart Environments), vol. 546. Cham, Switzerland: Springer, 2014, pp. 169–186.

[15] D. Pfisterer *et al.*, "SPITFIRE: Toward a semantic Web of things," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 40–48, Nov. 2011.

[16] *IDC FutureScape: Worldwide Internet of Things Predictions*. Accessed: Apr. 30, 2017. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=259856

[17] G. D. Abowd *et al.*, "Prototypes and paratypes: Designing mobile and ubiquitous computing applications," *IEEE Pervasive Comput.*, vol. 4, no. 4, pp. 67–73, Oct. 2005.

[18] K. Lyytinen and Y. Yoo, "Issues and challenges in ubiquitous computing: Introduction," *Commun. ACM*, vol. 45, no. 12, pp. 62–65, Dec. 2002.

[19] T. Verbelen, P. Simoens, F. De Turck, and B. Dhoedt, "Cloudlets: Bringing the cloud to the mobile user," in *Proc. MCS*, 2012, pp. 29–36.

[20] L. Gkatzikis and I. Koutsopoulos, "Migrate or not? Exploiting dynamic task migration in mobile cloud computing systems," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 24–32, Jun. 2013.

[21] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. MCC*, 2012, pp. 13–16.

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

[23] Research Councils UK. (2016). *GW4 Tier 2 HPC Centre for Advanced Architectures*. [Online]. Available: http://gtr.rcuk.ac.uk/projects?ref=EP%2FP020224%2F1

[24] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014.

[25] J. Balasangameshwara and N. Raju, "Performance-driven load balancing with a primary-backup approach for computational grids with low communication cost and replication cost," *IEEE Trans. Comput.*, vol. 62, no. 5, pp. 990–1003, May 2013.

[26] C. Dsouza, G.-J. Ahn, and M. Taguinod, "Policy-driven security management for fog computing: Preliminary framework and a case study," in *Proc. IRI*, Aug. 2014, pp. 16–23.

[27] L. Gavrilovska and R. Prasad, "Security in ad hoc networks," in *Ad-Hoc Networking Towards Seamless Communications* (Signals and Communication Technology). Amsterdam, The Netherlands: Springer, 2006, pp. 211–249.

[28] C. Babcock. (2014). *9 Worst Cloud Security Threats*. [Online]. Available: http://www.informationweek.com/cloud/infrastructure-as-a-service/9-worst-cloud-security-threats/d/d-id/1114085n?pagenumber=1

[29] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *Proc. FedCSIS*, Sep. 2014, pp. 1–8.

[30] L. Zhou and Z. J. Haas, "Securing ad hoc networks," *IEEE Netw.*, vol. 13, no. 6, pp. 24–30, Nov. 1999.

[31] A. Wasef, R. Lu, X. Lin, and X. Shen, "Complementing public key infrastructure to secure vehicular ad hoc networks [security and privacy in emerging wireless networks]," *IEEE Wireless Commun.*, vol. 17, no. 5, pp. 22–28, Oct. 2010.

[32] L.-C. Li and R.-S. Liu, "Securing cluster-based ad hoc networks with distributed authorities," *IEEE Trans. Wireless Commun.*, vol. 9, no. 10, pp. 3072–3081, Oct. 2010.
[33] R. Lacuesta, J. Lloret, M. Garcia, and L. Peñalver, "A Secure Protocol for Spontaneous Wireless Ad Hoc Networks Creation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 4, pp. 629–641, Apr. 2013.
[34] Q. Zhang, L. T. Yang, and Z. Chen, "Privacy preserving deep computation model on cloud for big data feature learning," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1351–1362, May 2016.
[35] Q. Zhang, H. Zhong, L. T. Yang, F. Bu, and Z. Chen, "PPHOCFS: Privacy preserving high-order CFS algorithm on the cloud for clustering multimedia data," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 12, no. 4s, p. 66:1–66:15, 2016.

YANG LIU received the B.Sc. and Ph.D. degrees from the Dalian University of Technology, China, in 2006 and 2010, respectively, and the M.Sc. degree from The University of Edinburgh, U.K., in 2007. From 2008 to 2015, he was a Lecturer with the Dalian University of Technology. Since 2015, he has been a Post-Doctoral Researcher in computer science with the University of Exeter, U.K. His research interests are in the areas of Internet-of-Things and fog computing.

JONATHAN E. FIELDSEND (S'00–M'02) received the B.A. degree (Hons.) in economics from Durham University in 1998, the M.Sc. degree in computational intelligence from Plymouth University in 1999, and the Ph.D. degree in computer science from the University of Exeter in 2003. He is currently an Associate Professor in computational intelligence with the University of Exeter. His research interests include multi-objective optimization, optimization with uncertainty, robust optimization, multi-modal optimization, pattern recognition, machine learning, and data visualization. He is a member of the IEEE Computational Intelligence Society (CIS) and the IEEE CIS Task Force on Multi-modal Optimization, a fellow of the Higher Education Academy, and sits on the South West Branch of the BCS. He is a Vice-Chair of the IEEE CIS Task Force on Data-Driven Evolutionary Optimization of Expensive Problems.

GEYONG MIN (M'–) received the B.Sc. degree in computer science from the Huazhong University of Science and Technology, China, in 1995, and the Ph.D. degree in computing science from the University of Glasgow, U.K., in 2003. He is currently a Professor of high performance computing and networking with the Department of Mathematics and Computer Science, College of Engineering, Mathematics and Physical Sciences, University of Exeter, U.K. His research interests include future Internet, computer networks, wireless communications, multimedia systems, information security, high performance computing, ubiquitous computing, and modeling and performance engineering.