Distributed Trade-Based Edge Device Management in Multi-Gateway IoT

FARZAD SAMIE, Karlsruhe Institute of Technology (KIT), Germany
VASILEIOS TSOUTSOURAS, National Technical University of Athens, Greece
LARS BAUER, Karlsruhe Institute of Technology (KIT), Germany
SOTIRIOS XYDIS and DIMITRIOS SOUDRIS, National Technical University of Athens, Greece
JÖRG HENKEL, Karlsruhe Institute of Technology (KIT), Germany

The Internet-of-Things (IoT) envisions an infrastructure of ubiquitous networked smart devices offering advanced monitoring and control services. The current art in IoT architectures utilizes gateways to enable application-specific connectivity to IoT devices. In typical configurations, IoT gateways are shared among several IoT edge devices. Given the limited available bandwidth and processing capabilities of an IoT gateway, the service quality (SQ) of connected IoT edge devices must be adjusted over time not only to fulfill the needs of individual IoT device users but also to tolerate the SQ needs of the other IoT edge devices sharing the same gateway. However, having multiple gateways introduces an interdependent problem, the binding, i.e., which IoT device shall connect to which gateway.

In this article, we jointly address the binding and allocation problems of IoT edge devices in a multigateway system under the constraints of available bandwidth, processing power, and battery lifetime. We propose a distributed trade-based mechanism in which after an initial setup, gateways negotiate and trade the IoT edge devices to increase the overall SQ. We evaluate the efficiency of the proposed approach with a case study and through extensive experimentation over different IoT system configurations regarding the number and type of the employed IoT edge devices. Experiments show that our solution improves the overall SQ by up to 56% compared to an unsupervised system. Our solution also achieves up to 24.6% improvement on overall SQ compared to the state-of-the-art SQ management scheme, while they both meet the battery lifetime constraints of the IoT devices.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; Sensor networks; • **Networks** → *Network resources allocation*; **Cyber-physical networks**;

Additional Key Words and Phrases: Internet-of-Things, IoT, edge computing, distributed resource allocation, computation offloading, constrained devices

ACM Reference format:
Farzad Samie, Vasileios Tsoutsouras, Lars Bauer, Sotirios Xydis, Dimitrios Soudris, and Jörg Henkel. 2018. Distributed Trade-Based Edge Device Management in Multi-Gateway IoT. *ACM Trans. Cyber-Phys. Syst.* 2, 3, Article 17 (June 2018), 25 pages.
https://doi.org/10.1145/3134842

This work was partially supported by the E.C. funded program AEGLE under H2020 Grant Agreement No: 644906.
Authors' addresses: F. Samie, L. Bauer, and J. Henkel, Haid-und-Neu-Str. 7, 76131, Karlsruhe, Germany; emails: {samie, bauer, henkel}@kit.edu; V. Tsoutsouras, S. Xydis, and D. Soudris, National Technical University of Athens (NTUA), School of Electrical & Computer Engineering, Department of Computer Science, Microprocessors, and Digital Systems Laboratory, 9 Heroon Polytechneiou, Zographou Campus, 15780 Athens, Greece; emails: {billtsou, sxydis, dsoudris}@microlab.ntua.gr.

https://doi.org/10.1145/3134842

1 INTRODUCTION

Recent advancements in technology in addition to emerging techniques in embedded systems, wireless communication, and sensors have enabled the design of small-size, ultra-low-power and low-cost IoT devices (Gubbi et al. 2013; Henkel et al. 2017). These devices use the network, or particularly the Internet, as an infrastructure for connecting to each other to communicate, coordinate, and cooperate in order to provide new services. This emerging paradigm is called the Internet of Things (IoT), which covers a wide range of applications including healthcare monitoring, smart city, smart buildings, and so forth (Gubbi et al. 2013; Miorandi et al. 2012; Samie et al. 2016a).

The IoT edge devices sense information from physical phenomena and send preprocessed data to a gateway node, which aggregates the streams of sensed data in real time and sends them to the central servers, e.g., fogs, clouldlets, or cloud servers, for storage or further analysis (Shi et al. 2016; Wu et al. 2011; Zhu et al. 2010). In some applications the data should be acted upon on the IoT device (i.e., devices have actuators), while other applications may just be used for monitoring (Samie et al. 2016a).

One of the challenges in IoT is to process and analyze a huge amount of data from heterogeneous devices. The massive number of IoT devices will lead to a rapid explosion of the scale of collected data (Chien et al. 2015). This challenge has two aspects: (1) the Big Data (Lee and Lee 2015) and (2) diverse application requirements of IoT (Zhang et al. 2015). Handling all these collected data with central cloud servers is inefficient, or sometimes even unfeasible, because of

—the limitation of computing, communication, and storage resources,
—the heavy load on the IoT network, and
—the unreliable latency (Salman et al. 2015; Want et al. 2015).

To address these issues, the task of processing the data is pushed to the network edges introducing the concepts of fog computing and edge computing (Ahmed and Ahmed 2016; Fernando et al. 2013; Salman et al. 2015). According to a report by IDC Futurescape, around 40% of IoT-generated data will be processed, stored, and acted upon close to the edge of the network (Raphael 2016). Edge computing enables analysis of information processing at the source of the data, which sometimes is referred to as in-network (He et al. 2015) or on-board processing (Lazarescu 2014). This paradigm not only reduces the huge workload of central computing servers (e.g., clouds) but also decreases the latency of data processing, which includes the latency for sending the required data plus the response time for performing the task on the cloud server (Samie et al. 2016b).

Many IoT devices are battery powered or have limited energy sources due to either their portability requirements or the lower cost of installation, deployment, and maintenance. This limitation imposes the following two constraints on IoT devices: (1) processing capability and (2) communication bandwidth. Due to the first constraint, some of the IoT applications are not able to autonomously process the entire collected data on the IoT devices. Hence, a portion of the processing task is *offloaded* to more powerful layers (e.g., gateways) (Bortolotti et al. 2016; Braojos et al. 2014). However, to offload the computation, the raw or partially/processed data must be transmitted to the gateway, and this is where the bandwidth constraint comes into play. The total throughput of low-power wireless technologies for IoT is only a few kilobits per second (Samie et al. 2016b), which sometimes may even decrease due to the interference.

IoT envisions a model in which the increasingly ubiquitous, portable, smart devices provide gateway services (Zhang et al. 2015). The gateway aims at (1) bridging the low-power wireless network of IoT devices and the Internet (Riliskis et al. 2015; Zhu et al. 2010) and (2) providing local processing to reduce the latency (compared to cloud servers) (Catarinucci et al. 2015; Zachariah et al. 2015). Some IoT local networks may exploit multiple gateways to cover an area or to support

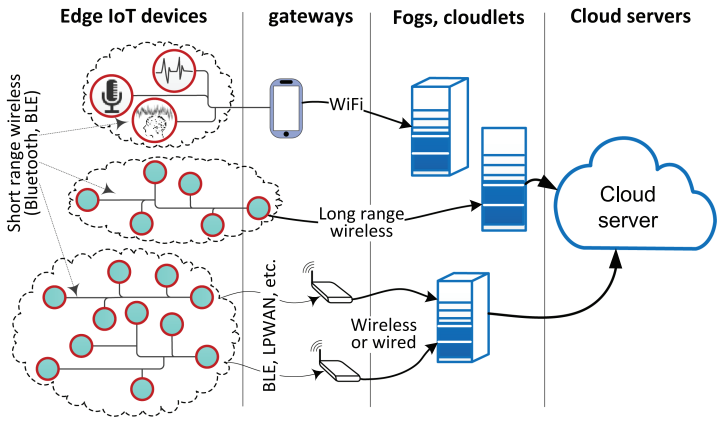


Fig. 1. Local networks of IoT devices connected to the Internet via gateways.

more IoT devices. Figure 1 illustrates the envisioned model in IoT with edge devices, multiple and heterogeneous gateways, fogs, and cloud storage/computing servers.

Battery-powered IoT devices need to fulfill a specific expected lifetime before the next recharge takes place. To manage the energy consumption under the battery lifetime constraints, each IoT device has generally two control knobs: (1) changing the service quality (SQ) and (2) changing the on-board processing policy. For instance, in an IoT-based health monitoring device that captures and transmits the electrocardiogram (ECG) signal to the Internet through a smartphone, when the battery level is low, the device can reduce the sampling rate of data acquisition from 360Hz to 250Hz. Alternatively (or conjointly), it can stop performing digital filtering for baseline wandering removal (Lee et al. 2011) and offload it to the gateway. However, the gateways are battery powered with limited energy sources too. They have limited resources for receiving data and limited processing capability of performing the offloaded computation. Depending on its remaining energy, the gateway should restrict its available bandwidth and processing power offered to other IoT devices. It should be noted that although the gateway might be equipped with a high-bandwidth connection to the Internet (e.g., WiFi), its interface with IoT devices is still a low-power wireless connection such as Bluetooth Low Energy (BLE), ZigBee, LoRa, and so forth, which have a low bandwidth (Al-Fuqaha et al. 2015; Hassanalieragh et al. 2015; Zachariah et al. 2015).

In such a local network, IoT devices aim at reaching the highest overall SQ while meeting their battery lifetime constraints. The existence of multiple gateways provides some IoT devices more than one option to connect and receive gateway service (we refer to it as the *binding problem*). The binding problem arises when the ranges of wireless radios of gateways overlap and there are some IoT devices in the overlapping areas. Therefore, those IoT devices have multiple choices as their gateway. The importance of efficient binding is to avoid situations where some gateways are overloaded while other gateways are underutilized. For instance, if several IoT devices with high data transmission demand are connected to the same gateway, they must reduce their sampling rates to meet the constraint on the limited shared communication bandwidth. Consequently, the overall SQ of applications decreases. The binding problem is dynamic and requires an online solution. Consider an example where some devices are mobile (i.e., users are moving in the building). As the location of an IoT devices changes, the gateways that can reach it will change too.

A mechanism is needed to capture the dynamism in an IoT network and help the IoT devices to choose (1) their gateway and (2) their best operational configuration using their control knobs

(i.e., SQ level and offloading scheme). These decisions are interdependent and should be made conjointly. SQ management under battery lifetime constraints in a multigateway IoT system needs dynamic online solutions for the following reasons:

(1) The remaining energy of IoT devices varies over time due to consumption or recharge.
(2) The available bandwidth or processing capability of the gateways may change over time according to their available power.
(3) The number of IoT devices connected to the gateway may change (Kim 2015) due to the mobility of the IoT devices (i.e., user) (Kliem and Kao 2015).

It necessitates an online mechanism to address efficient SQ management and offloading in such a dynamic IoT configuration.

The novel contributions of this work are as follows:

—We present a resource management scheme for multigateway IoT systems with constraints on battery that jointly addresses efficient binding, bandwidth allocation, and computation offloading.
—We analyze in-depth the new problem instance and suggest mechanisms to enable fine-grained SQ levels to be achieved.
—We propose a distributed agent-based mechanism that improves the overall SQ by trading IoT devices between gateways.

The rest of this article is organized as follows. In Section 2, we review related work and background on computation offloading and edge processing as well as SQ management in IoT, WSN, and mobile network systems. After that, we present the problem formulation in Section 3, and then we provide a detailed presentation of our proposed solution in Section 4. Experimental results and evaluations are presented in Section 5, while we conclude the article in Section 6.

2 BACKGROUND AND RELATED WORK

Although SQ management and computation offloading in edge computing are among the foreseen challenges in IoT (Christin et al. 2009), there are not many research works addressing the joint problem. However, the problems of computation offloading, SQ management, and quality of experience (QoE) have been addressed separately in other research fields including WSN (Barbancho et al. 2006; Chen and Varshney 2004; Xia 2008) and mobile computing (Kumar et al. 2013; Li et al. 2001; Wang et al. 2015).

Many research efforts have been conducted on computation offloading in mobile systems (Kumar et al. 2013). Although in these systems mobile devices are battery powered and resource constrained, the destination of offloading (i.e., servers) is powerful and resourceful computers.

MiLAN (Heinzelman et al. 2004; Murphy and Heinzelman 2003) is middleware to manage and allocate network resources for applications that *fuses* data from multiple sensors and needs to select an optimal set of sensors. However, the SQ of each sensor is fixed. Moreover, MiLAN only considers the bandwidth limitation of the network, while the processing capability is not modeled. Besides, it does not model the on-board processing and therefore cannot support combinations of offloading schemes.

Sheng et al. (2015) propose to minimize the energy consumption of sensor nodes by computation offloading. This approach considers an IoT device solely to find the optimal partitions of its application for computation offloading. The output of this approach can be used as the input to our problem.

A decentralized game-theoretic approach for computation offloading in mobile computing systems is presented in Chen (2015). A single gateway is considered, whose wireless channel is the scarce and shared resource, while the processing capability of the server is unlimited. In addition, this approach does not support multiple levels of offloading for a device. In other words, the decision is between fully offloading the computation and fully processing on-board.

A joint optimization of bandwidth and computational resources for computation offloading in a dense deployment scenario is presented in Sardellitti et al. (2015), which considers the presence of radio interference. The proposed solution is, however, for single-gateway networks, and it does not support multiple SQ levels of devices either. In Xian et al. (2007) and Kovachev et al. (2012), adaptive approaches are proposed to offload the computation from portable devices to extend the battery lifetime.

Other research works have been proposed to dynamically offload the computation (Huang et al. 2012; Kwak et al. 2015; Mao and Zhang 2016). However, they aim at a system with single-gateway (server) architecture. In addition, they consider each device (or user) solely, thus not addressing the effect of different devices on each other by consuming the limited shared resources.

In Oddi et al. (2015), an approach based on game theory is proposed to allocate the bandwidth dynamically in a shared network channel to manage the quality of experience. Although the algorithm can be adjusted for a single-gateway IoT system to allocate the bandwidth, it does not support any computation offloading aspect. In addition, it cannot be extended to a multigateway system.

Xia et al. (2007) proposed a feedback control scheme to manage the quality of service (QoS) of sensor nodes in a WSN. The QoS manager adapts the sampling rate (i.e., SQ) of sensors at runtime based on observed parameters such as transmission delay or packet loss. However, the target system does not incorporate on-board processing (i.e., it always offloads all the gathered data). Packet loss is allowed to happen and then the system reacts to it by adjusting the SQ level of the sensor nodes. Note that even though QoS is commonly used in networking literature to refer to the aspects of network service (e.g., throughput, delay, etc.), it may cover a wider range of parameters, especially in the IoT domain (Li et al. 2014). For instance, the quality of the final results that the user receives (i.e., SQ) is a QoS parameter.

In Ma et al. (2007), a utility-based approach is studied for bandwidth allocation in wireless networks. The proposed approach is market oriented but in a centralized fashion that is designed for single-gateway systems. Hence, it is not applicable to multigateway systems. It does not support computation offloading either. Another protocol for bandwidth allocation is presented in Chen and Heinzelman (2005) for mobile ad hoc networks. It uses a method to estimate the approximate bandwidth, finds the residual available bandwidth, and then reacts to the network traffic. This solution is not applicable to our problem, where applications operate at different input data rates with multiple offloading policies. It does not support multiple gateways either.

We studied the joint problem of bandwidth allocation and SQ management under resource constraints for a single-gateway IoT system in Samie et al. (2016c). The optimal allocation bandwidth and computation offloading level is determined using a dynamic programming algorithm. Given a binding (i.e., which IoT device is connected to which gateway), the proposed algorithm can be used to find an efficient resource allocation. However, in a multigateway system, the problem has two interdependent subproblems: binding and allocation.

In summary, none of the presented approaches is applicable to our problem for the following reasons: (1) They assume a resourceful cloud server for the destination of offloading while in our system the computation is offloaded from IoT edge nodes to other battery-operated devices with limited processing power. (2) They consider the devices solely and assume that the offloading policy of one device does not affect the other devices in their setup. However, in our problem

several IoT devices share the limited resources of gateways. Thus, the resource allocation for one device would affect the other devices. (3) The binding problem does not exist in their setup as they are not involved with multiple destinations for offloading. (4) None of the aforementioned related works support systems where portable devices provide different SQ levels.

3 SYSTEM MODEL AND PROBLEM FORMULATION

3.1 Application Model

An IoT application offers its service at different qualities, ranging from "low" to "high." Higher service quality increases the user's satisfaction but at the cost of more resource usage (e.g., energy consumption, bandwidth, etc.). Different service qualities usually derive from different qualities of input data. For instance, consider an IoT-based heart monitoring device that can capture ECG signals at multiple discrete sampling rates (e.g., 100Hz, 200Hz, 300Hz, and 1,000Hz). The quality of the captured signal determines the service quality and affects the user's experience and satisfaction.

Moreover, the IoT application may either (1) fully process the captured data on the IoT device and send only the results to the Internet, (2) partially process the data on the IoT device and offload the rest of the computation to the gateway, or (3) offload the whole computation to the gateway by transmitting the raw data. This would lead to different possible offloading levels. For instance, consider that the aforementioned IoT-based heart monitoring device had three offloading levels. Level 1 could indicate "no offloading" and thus only transmit a small amount of results (e.g., the features extracted from the signal), independently of the input data rate. Level 2 could be used to offload a certain percentage of the input sample rate. For instance, the device could store a certain amount of input samples in a buffer, then process the buffer and offload all incoming samples during this processing to the gateway. Level 3 could perform some preprocessing on the data, e.g., apply a filter on it, and then send the filter output to the gateway. As the filter would not reduce the data rate, the transmission rate would equal the input data rate, but some of the processing is already done.

3.2 IoT Device Model

We consider an IoT network consisting of N portable IoT devices, where each device I_d, $d \in \{1, \ldots, N\}$ is described by a tuple:

$$I_d = (X_d, R_d, B_d, e_d, U_d(\cdot), C_d(\cdot)), \tag{1}$$

where

— X_d denotes the set of possible *input data rates* of device d. They depend on the sensor sampling frequency and data resolution. An IoT device offers its service at M_d different *SQ levels*, with each level having a different input data rate and thus providing a different service quality:

$$X_d = \{x_{d_i} \mid i \in [1, M_d]\}. \tag{2}$$

— R_d denotes the set of possible *transmission data rates* of device I_d. They depend on the input data rate x_{d_i} and the computation offloading strategy of the IoT device. Offloading determines how much input data is not processed on the device (on-board processing) but transmitted to the gateway (offloaded) instead. An IoT device offers Q_d different *offloading levels*. The data transmission rate $r_{d_{ij}}$ depends on the SQ level i (input data rate) and the offloading level j. It corresponds to the share of the input data rate that is offloaded plus the (intermediate) results from the on-board processed data:

$$R_d = \{r_{d_{ij}} \mid i \in [1, M_d],\ j \in [1, Q_d]\}. \tag{3}$$

The particular transmission data rates depend on the device and how it is used, which has to be determined by the user and thus is considered as given in this problem formulation.
- B_d denotes the minimum required battery lifetime (i.e., until the next recharge or battery replacement).
- e_d is the remaining energy in the battery of device d.
- $U_d(x_{d_i})$ is the utility function that quantifies the utility or SQ provided to the user when the device is capturing input data at rate x_{d_i}.
- $C_d(i, j)$ is the total power consumption for sensing and capturing input data at rate x_{d_i}, processing it under offloading level j, and transmitting the data at rate $r_{d_{ij}}$. It includes the power consumption for sensing, computation, and communication.

The battery lifetime of IoT devices depends on (1) the remaining energy and (2) the total power consumption rate:

$$b_{d_{ij}} = \frac{e_d}{C_d(i, j)}, \tag{4}$$

where $b_{d_{ij}}$ denotes the expected battery lifetime when the device captures input data at rate x_{d_i}, processes it, and then transmits at rate $r_{d_{ij}}$.

3.3 Gateway Model

The gateway connects devices to the Internet. It receives data from IoT devices, processes it, and transmits the final result to the Internet.

We consider a set of G gateways where each gateway g is specified by a tuple:

$$\text{gateway } g: \ (p_g(\cdot), R_g, P_g),$$

where

- $p(r_{d_{ij}})$ shows the required processing capability of the gateway to perform the necessary operations on the received data at rate $r_{d_{ij}}$ and offloading level j,
- R_g is the total available bandwidth of the gateway to receive data from IoT devices, and
- P_g shows the total processing capability of the gateway.

The effect of the environment and surrounding devices (e.g., wireless interference) on the transmission can be modeled in the gateways' parameter, R_g, and the IoT devices' parameter, $C_d(\cdot)$. For instance, if an IoT device observes an increase in the re-transmission rate, it can increase the cost of transmission. However, modeling the effect of interference on the transmission parameters is beyond the scope of this work.

3.4 Network Model

While the gateways are assumed to be stationary and nonmobile, the IoT devices are what we call *quasi mobile*. The IoT devices are mobile but their location does not change significantly and frequently, which results in low mobility. For instance, the IoT devices used for patient monitoring in a smart hospital or smart home have a mobility pattern of the natural movement of patients (López et al. 2010).

Depending on the location of IoT devices and gateways, each IoT device may reach some gateways (at least one). Each IoT device should connect to one and only one gateway in order to transmit the data/results to the Internet through it. For those IoT devices that reach multiple gateways, the binding needs to be decided. Let matrix $A[\cdot]_{N\times G}$ show which IoT devices reach which gateways:

$$A[dg] = \begin{cases} 1 & \text{if device } d \text{ reaches gateway } g \\ 0 & \text{otherwise.} \end{cases}$$

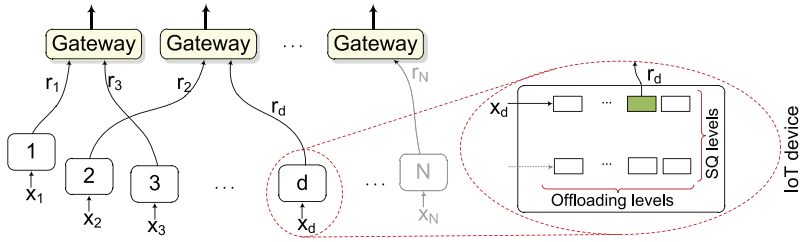


Fig. 2. Problem model: IoT devices with different SQ and offloading levels resulting in different transmission data rates. Multiple gateways receive and process the data.

3.5 Problem Statement

The system is summarized in Figure 2. The problem that is targeted in this article can be solved by (1) deciding the binding of IoT devices to the gateways and (2) choosing the SQ level i and the offloading level j for each IoT device d at runtime, such that the bandwidth, computation, and lifetime constraints are fulfilled (Equation 5–7) and the overall benefit (Equation (8)) is maximized:

$$\text{Bandwidth constraint:} \quad \sum r_{d_{ij}} \leq R_g \qquad \forall d \text{ connected to } g \tag{5}$$

$$\text{Computation constraint:} \quad \sum p(r_{d_{ij}}) \leq P_g \qquad \forall d \text{ connected to } g \tag{6}$$

$$\text{Lifetime constraint:} \quad \forall\, d:\ b_{d_{ij}} \geq B_d \tag{7}$$

$$\text{Optimization goal:} \quad \text{maximize} \sum_{\forall d} U_d(x_{d_i}). \tag{8}$$

4 PROPOSED SOLUTION

4.1 Decomposing the Problem

The targeted problem has two sets of constraints: one for IoT devices and one for gateways. The selected configurations for devices d (i.e., x_{d_i} and $r_{d_{ij}}$) should meet the lifetime constraint (Equation (7)). Given the selected configuration for IoT devices, the gateway's constraints to meet are the bandwidth and computation (Equation 5, 6).

The device's constraint depends solely on device parameters. To reduce the search space, we can decompose the optimization problem into (1) the device's problem and (2) the network (gateway's) problem. In the device's problem, each IoT device excludes those configurations that violate its lifetime constraint to reduce the search space. Then, the network (gateway's) problem is solved by considering the reduced search space.

4.2 Device Problem: Addressing Battery Lifetime Constraint

Considering its lifetime constraint, each device has a set a valid configurations. It finds the efficient feasible configurations (EFCs), each of which corresponds to an SQ level with the minimum data transmission rate. Each EFC is a pair containing (1) the utility and (2) the transmission data rate r_{d_i} of this configuration. The gateway extends each EFC set by including the processing requirement of the associated transmitted data (i.e., $p(r_d)$). The resulting EFC'_d set is shown in Equation (9). Note that the latency constraint is important in some IoT applications. We assume that each individual IoT device takes care of its latency constraint by providing the EFC set that fulfills this constraint. If one configuration does not satisfy the real-time constraint, the device excludes it from its EFC set, and naturally, this configuration will not be selected for this device. The evaluation of the constraints (latency, battery lifetime) is online and dynamic: the device can

monitor and adjust the parameters to include/exclude the configurations at runtime. The details on finding the EFC set can be found in Samie et al. (2016c):

$$\begin{aligned} EFC'_d &= \left\{ (U_{df},\ r_{df},\ p_{df})_{1 \le f \le |EFC_d|} \right\} \\ U_{df} &= U_d(x_{d_f}) && \text{// Utility of device } I_d \text{ for the } f\text{th EFC entry} \\ r_{df} &= r_{d_f} && \text{// Corresponding transmission rate} \\ p_{df} &= p(r_{d_f}) && \text{// Corresponding gateway processing requirement.} \end{aligned} \tag{9}$$

Each IoT device periodically checks its remaining energy e_d and updates the EFC set. In case the EFC set changes, the device sends the new set to its gateway, where it is used to solve/update the *network problem*. Note that the EFC set only contains feasible solutions; i.e., the number of entries in the EFC set of a particular device may change over time.

4.3 Network or Gateway's Problem

Given the EFC sets of IoT devices, the Efficient Multigateway Allocation and Binding (EMGAB) problem can be formulated as

$$\max \quad \sum_d \sum_f (U_{df} \times w_{df}) \tag{10}$$

$$\text{subject to} \quad \forall d: \quad \sum_f w_{df} = 1 \tag{11}$$

$$\forall d: \quad \sum_g v_{dg} = 1 \tag{12}$$

$$\forall d, g: \quad v_{dg} \le A[dg] \tag{13}$$

$$\forall g: \quad \sum_d \sum_f r_{df} \times w_{df} \times v_{dg} \le R_g \tag{14}$$

$$\forall g: \quad \sum_d \sum_f p_{df} \times w_{df} \times v_{dg} \le P_g, \tag{15}$$

where

$$w_{df} = \begin{cases} 1 \text{ if the } f\text{th EFC element from the } d\text{th device is chosen} \\ 0 \text{ otherwise.} \end{cases} \tag{16}$$

$$v_{dg} = \begin{cases} 1 \text{ if the } d\text{th device is connected to the } g\text{th gateway} \\ 0 \text{ otherwise.} \end{cases} \tag{17}$$

Equation (11) ensures that one configuration for each device is selected. Equation (12) and (13) ensures that each IoT device is connected to one gateway that is in its range. Finally, Equation (14) and (15) ensures that the binding of IoT devices to the gateways and their selected configurations meet the constraints of each gateway.

4.4 Analysis of the Problem

As stated in Section 3.4, some IoT devices have multiple choices to connect to gateways, while some others reach only one gateway (i.e., no decision is needed for binding, but they still need to choose the SQ level and offloading policy). Let H_d denote the set of gateways reachable by device d ($|H_d| \ge 1$). The total number of bindings to investigate is $\Pi_{d=1}^{N} |H_d|$. Then, for a possible binding setup, there are G optimization problems to find the optimal SQ level and offloading policy for the IoT devices. The optimal solution for one instance (i.e., a single gateway) is presented in Samie et al. (2016c).

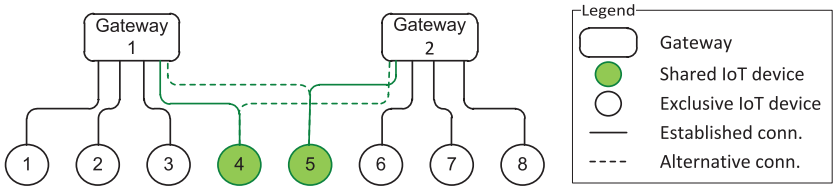


Fig. 3. An example with two gateways sharing two IoT devices while each has three exclusive IoT devices.

Lemma 4.1. *The EMGAB problem (presented in Equation 10–17) is strongly NP-complete.*

Proof. The numerical parameters of EMGAB (i.e., N, X_d, and R_d) are bounded by a polynomial. The reason is the limitation of practical setup in IoT systems. Now we show that for the bounded parameters, the EMGAB problem remains NP-complete: EMGAB is a generalization of the Multiple Choice Multidimensional Multiple-Knapsack Problem (M^3PK). Each gateway corresponds to a knapsack that has two constraints: bandwidth corresponds to the "volume" and processing power corresponds to the "weight." Each EFC set of a device corresponds to a class of items among which one item must be picked. This transformation between EMGAB and M^3PK is done in polynomial time. The M^3PK problem is a generalization of the Multiple Knapsack Problem (MKP), Multiple Choice Knapsack Problem (MCKP), and Multiple Choice Multidimensional Knapsack Problem (MMKP), which are proven to be NP-complete. Hence, an NP-complete problem is reduced to the EMGAB in polynomial time. Due to the polynomial bound on the inputs, the EMGAB problem belongs to the class of strongly NP-complete problems. □

The problem is computationally difficult to solve in a centralized fashion. Besides, the nature of IoT systems, with multiple devices and multiple gateways, is distributed. Therefore, distributed or decentralized strategies are promising solutions that take autonomous decisions for IoT devices and gateways based either on local information or on an incomplete picture of the global network status. Market-oriented approaches are usually used for distributed resource allocation problems and can be classified into three models: (1) price based, (2) auction based, and (3) trade based (Smolinski 2000). Mechanisms that are based on price and auction usually require a centralized entity with a full picture of network conditions (e.g., an auctioneer to run the auction or a decision maker to calculate the price) (Kim 2015; Ma et al. 2007; Smolinski 2000). Due to the nature of the system and problem, a trade-based distributed solution seems more effective.

4.5 Distributed Solution and MGAB Protocol

In this subsection, we present a distributed solution to the MGAB problem (i.e., multigateway allocation and binding) and the detailed protocol to implement it. Our solution is based on trading, in which gateways are modeled as intelligent agents that negotiate with each other to *take*, *give*, or *exchange* the connected IoT devices.

Definition 4.2. Common versus exclusive IoT devices: From the perspective of gateways, an IoT device is either reachable by only one gateway, which is called *exclusive*, or reachable by more than one gateway, which is then *common* between them. Figure 3 shows an example with two gateways sharing two common IoT devices (i.e., devices 4 and 5) and three additional exclusive IoT devices per gateway. In the current *established connections*, IoT nodes 4 and 5 are connected to gateways 1 and 2, respectively. However, there are three other *alternative connections* for them.

While the only action for exclusive nodes is to change the SQ and/or offloading level, there are two extra actions for common nodes:

(1) Migration: One common node leaves its current gateway and joins the other one. For instance, if IoT device ④ disconnects from gateway 1 and connects to gateway 2, it is called migration.
(2) Exchange: The two gateways exchange two common IoT devices with each other. For instance, IoT device ④ connects to gateway 2 in return of ⑤ being connected to gateway 1.

Given a binding/allocation setup, there are two situations that make it inefficient and an *exchange* or *migration* could address them.

Definition 4.3. Fragmentation: This is a phenomenon in which gateways have unused resources that might not be enough for increasing the SQ level of the current connected IoT devices, but might be enough for increasing the SQ level of a common IoT device that is currently connected to the other gateway.

Definition 4.4. Heterogeneity of resource usage: The IoT devices might be different in terms of resource usage: some might tend to use more bandwidth, while some might tend to use more processing power. This can result in a situation where one gateway has much unused bandwidth and the other has much unused processing power. We refer to this phenomenon as heterogeneity in resource usage.

4.6 Properties of Applications and Problem

We drive and introduce some properties of our applications and the considered system that can be exploited to reduce the complexity of the problem.

Property 1. *Considering device d and two elements of its EFC'_d set, (U_{di}, r_{di}, p_{di}) and (U_{dj}, r_{dj}, p_{dj}): If $U_{di} > U_{dj}$, then at least one of these conditions holds: (1) $r_{di} > r_{dj}$ or 2) $p_{di} > p_{dj}$. The rationale behind it is that otherwise the ith element dominates the jth element and is always preferable to it.*

Property 2. *Considering device d and two elements of its EFC'_d set, (U_{di}, r_{di}, p_{di}) and (U_{dj}, r_{dj}, p_{dj}): The device d can operate in a way to provide any average utility (SQ) U'_d, $U_{di} \leq U'_d \leq U_{dj}$, from the gateway's and application's perspective.*

Proof. Consider a constant time interval of T. If the device operates at the jth point for t_1 time and then changes the SQ level and operates at the ith point for $(T - t_1)$ time, then the average utility, transmission rate, and processing power usages are

$$U'_d = \frac{t_1 \times U_{dj} + (T - t_1) \times U_{di}}{T} \quad r'_d = \frac{t_1 \times r_{dj} + (T - t_1) \times r_{di}}{T} \quad p'_d = \frac{t_1 \times p_{dj} + (T - t_1) \times p_{di}}{T}. \quad (18)$$

□

When a device is supposed to deliver an intermediate SQ level (e.g., U'_d in Equation (18)), it uses the following scheme: It starts operating at the jth configuration (which has higher utility and consumes more resources on the gateway compared to the ith configuration). It keeps working at this configuration for t_1 time. However, it transmits its data to the gateway at the rate of r'_d ($r'_d \leq r_{dj}$). It buffers the rest of the produced data (i.e., $r_{dj} - r'_d$) on the memory. Then after t_1, it switches to the ith configuration that produces data at the rate of r_{di} ($r_{di} \leq r'_d$). It still transmits the data to the gateway at the rate of r'_d, which consists of previously buffered data and newly generated data. It keeps operating at the ith configuration for $(T - t_1)$ time and then repeats this procedure as long as the requested utility remains U'_d. Therefore, while the device is only operating at the ith and jth configurations and switching between them, the gateway sees the device operating at an intermediate configuration mode with (U'_d, r'_d, p'_d).

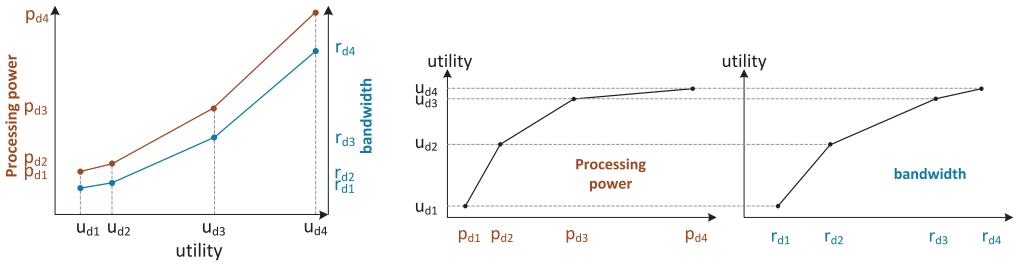


Fig. 4. The extended utility function of device d derived from discrete EFC'_d set. The function is piecewise linear and weakly concave with respect to both variables (i.e., r and p).

In summary, this new operating point (i.e., configuration) is the application-level function of the device. The device still operates only in discrete configurations, but it is transparent to the gateway, and the average effect from the perspective of application and gateway is continuous.

Forming Piecewise-Linear Utility Function: Using Proposition 2, we can expand the discrete utilities of each device to a piecewise-linear function. However, since each discrete utility value (e.g., U_{di}) corresponds to two variables (e.g., r_{di} and p_{di}), there will be two uni-variable utility functions. These two variables are dependent. Figure 4 shows an example of these two utility functions for device d.

Property 3. Concavity of utilities *is a general property of typical applications (Bhattacharya et al. 2010; Ma et al. 2007). It is a natural restriction based on the "law of diminishing returns" from economic concepts (Cho and Goel 2006; Goel and Nazerzadeh 2014).*

According to Proposition 2, 3, the piecewise-linear utility functions are *weakly concave*. In the example of Figure 4(b), the concavity of utility functions is illustrated.

Claim 1. *On each gateway that exploits Proposition 2, when the highest overall utility is reached, either at least one of the resources is fully utilized or the IoT devices are operating at their highest SQ level.*

Proof. Suppose for the purpose of contradiction that gateway g reaches the highest overall utility but it still has $\hat{R} > 0$ and $\hat{P} > 0$ resources left, **and** there is one connected IoT device, d, which is not working at its highest SQ level (i.e., U_{d_M}). Let's assume that this device is operating at the ith SQ level, $(i < M_d)$, which is the ith element in its EFC' set (i.e., $(U_{d_i}, r_{d_i}, p_{d_i})$). Its next SQ level is associated with the $(i+1)$-th element of its EFC' set described by $(U_{d_{i+1}}, r_{d_{i+1}}, p_{d_{i+1}})$. If $r_{d_{i+1}} \leq \hat{R}$ and $p_{d_{i+1}} \leq \hat{P}$, then we are able increase the SQ one level to the $(i+1)$-th level, which is a contradiction. Therefore, at least one of these conditions hold: (1) $r_{d_{i+1}} > \hat{R}$, or (2) $p_{d_{i+1}} > \hat{P}$. Let's assume that the first condition holds, and then

$$\exists\, 0 < t_1 < 1 : \quad t_1 = \frac{r_{d_{i+1}} - \hat{R}}{r_{d_{i+1}} - r_{d_i}}. \tag{19}$$

If for each unit of time the device operates t_1 portion at the ith SQ level and $(1 - t_1)$ portion at the $(i+1)$-th SQ level, it can provide $U' = t_1 \times U_{d_i} + (1 - t_1) \times U_{d_{i+1}}$ utility, which clearly is higher than U_{d_i}. This too contradicts the assumption. The same argument can be presented for the second condition (i.e., $p_{d_{i+1}} > \hat{P}$). If both conditions hold, the minimum t_1 value is selected. To summarize, by using Proposition 2, we are able to increase the overall utility, which is a contradiction. □

Proposition 1. *According to Claim 1, we can address the fragmentation problem using Proposition 2.*

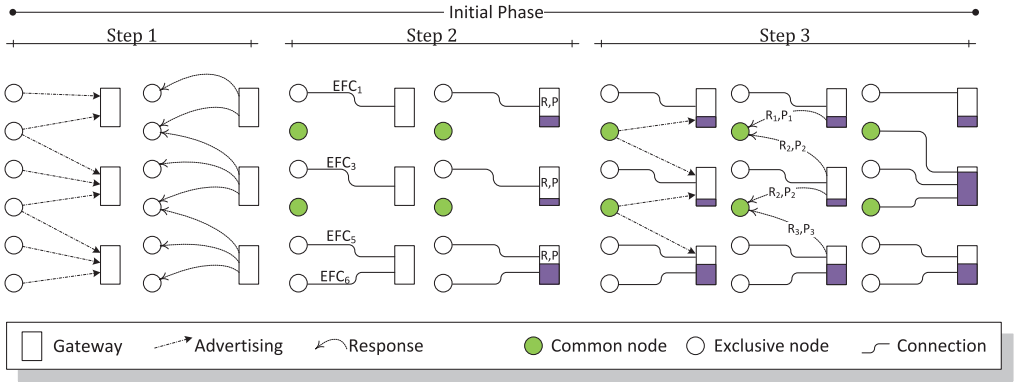


Fig. 5. Different steps during the initial phase, followed by trade phase.

The fragmentation problem stems from discrete and coarse-grained configurations (i.e., EFC elements), which can be addressed by continuous utility functions derived using Proposition 2.

4.7 Details of Agent-Based Approach

After addressing the fragmentation problem, we need to deal with the heterogeneity problem. In the following, we present the details of our distributed solution, which is an agent-based negotiation mechanism between gateways, for migration and exchange of IoT devices. It starts with an initial setup and then converges step by step to the efficient solution by increasing the overall utility of IoT devices. Each gateway is modeled as an autonomous intelligent agent, where the interests (or goals) of each agent is consistent with the goals of the whole MGAB problem. We use the terms "agent" and "gateway" interchangeably from here on.

4.7.1 Initial Phase. During the initial phase, gateways try to establish connections with IoT devices to reach the first setup. Figure 5 shows an example for the initial phase with six IoT nodes and three gateways. It consists of three steps:

(1) Advertisement and discovery: First, the IoT devices broadcast advertising packets to find the gateways. Each gateway sends requests to establish the connection, one for each IoT device in its range (in response to the advertisement packets). All exclusive IoT devices receive only one request (as they reach only one gateway); others receive more than one request.
(2) Exclusive connections: Then, each exclusive IoT device accepts the request of its gateway and connects to it. It sends its EFC set to the gateway. After establishing the connection and receiving the EFC set of devices, the gateway checks the lowest SQ level of connected devices and calculates the remaining resources to make sure that it still has enough resources to provide service to new devices. If not, it should stop sending requests to other IoT devices and accepting new devices.
(3) Other connections: Finally, common IoT devices send new advertisement packets. In return, the gateways in range send updated requests along with the information on their remaining resources. Each common IoT device accepts the first request for connection that has enough resources to support its lowest SQ level.

The advertisement and discovery mechanism is based on the BLE protocol where slave nodes (IoT devices) initiate the advertising and the master node (gateway) sends the request in response

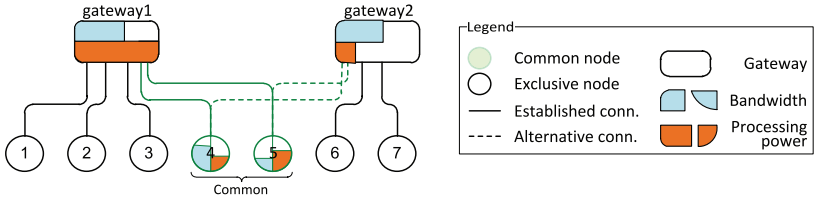


Fig. 6. Heterogeneity in resource usage of IoT devices and gateways.

to the advertisement (Townsend et al. 2014). However, other protocols or wireless technologies (e.g., ZigBee) can adopt a similar mechanism.

Once the initial phase ends, each gateway finds the optimal setup (SQ level and offloading policy) for its currently connected IoT devices based on the received information from its devices (as presented in Samie et al. (2016c, 2016b)).

4.7.2 Trade and Negotiation Phase. After the initial phase, each IoT node is bound to one gateway. We refer to this setup as "initial binding." Each gateway finds the optimal SQ level for its connected IoT devices. Since the number of connected devices to a gateway is practically small and limited, the gateway can find the optimal solution by executing a brute-force search or alternatively using faster algorithms similar to the solution proposed in Samie et al. (2016c). The resulting configuration is only optimal for the current binding (i.e., single-gateway level), but still not optimal for the whole network. We call this the initial allocation. It is optimal for the initial binding.

The initial phase is followed by a trade and negotiation phase, where agents talk to each other to adapt the binding and move it toward the optimal binding. This is done by a *market-based* approach, where agents make offers for exchanging or migrating the connected IoT devices.

In the negotiation phase, two agents are involved in a trade: an agent initiates a trade by sending a request either to migrate one of its shared devices to the other agent or to exchange a shared device with another one. The other agent, after doing some evaluations, may (1) accept the offer, (2) reject the offer, or (3) make another offer in response to the request. This decision is in the direction of increasing the overall SQ of the system. In other words, each trade should be a *Kaldor/Hicks improvement* (Coleman 1979), where those IoT devices that are made better off gain more than what other IoT devices that are made worse off lose.

A. Migration

The migration is first and mainly meant to address the uneven binding (i.e., overloaded gateways). However, it can address the heterogeneity issue too. First, each gateway, g, obtains two parameters: $\hat{R}_g$ and $\hat{P}_g$, which denote the amount of bandwidth and processing power left, respectively. Note that these values are nonzero for the gateways whose connected nodes are at their highest SQ level (see Claim 1). The migration is feasible only if one gateway (i.e., source) has no resources left (i.e., $\hat{R}_g \times \hat{P}_g = 0$), but the other gateway (i.e., destination) still has available resources.

We consider the example shown in Figure 6 and explain the negotiation of two agents using it. In this example, nodes 4 and 5 are common between gateway 1 and gateway 2. In the initial binding, both nodes are connected to gateway 1. The trade and negotiation details of a migration are as follows:

(1) **Picking the candidate to migrate:** Gateway 1 selects one of its common (i.e., nonexclusive) IoT devices for migration. It gives the highest priority to the node that consumes the scarcer resource more (i.e., makes the heterogeneity issue worse). For instance, in

Figure 6, node 5 consumes more processing power, which is the scarcer resource on gateway 1. Therefore it is prioritized over node 4.

(2) **Checking the usefulness of migration:** Gateway 1 calculates the 3-tuple of $(\overset{+}{u}, \overset{+}{r}, \overset{+}{p})$. This triple describes the best possible improvement among IoT devices *if* the candidate common device is migrated to another gateway. Therefore, $\overset{+}{u}$ is the gained utility, while $\overset{+}{r}$ and $\overset{+}{p}$ show the remaining bandwidth and processing power of gateway 1 after this improvement, respectively. For instance, if node 5 migrates from gateway 1 to gateway 2, then other nodes might be able to increase their SQ level at the cost of getting more bandwidth and processing power from gateway 1. If the candidate device migrates to another gateway, it releases its resources on the first gateway, freeing up r^* and p^* bandwidth and processing power, respectively.

(3) **Make an offer:** Gateway 1 sends an offer of migration to gateway 2, which includes the information about (1) its benefit from improvement (i.e., $\overset{+}{u}$), (2) the EFC set of the candidate node, and (3) the current operating configuration of the candidate node (i.e., (u_5^*, r_5^*, p_5^*) in our example).

Upon receiving this offer (i.e., migration request), the destination gateway (i.e., gateway 2) evaluates the offer and replies to it. The offer is evaluated to check if this move can improve the overall SQ of the system. The outcome of this evaluation determines whether to accept or reject the offer. The offer evaluation has two cases:

(1) First, it compares its remaining unused resources (i.e., $\hat{R}_2$ and $\hat{P}_2$) with the requirements of the offered candidate. If it has enough unused resources to host the migrated device, i.e., $\hat{R}_2 \geq r^*$ and $\hat{P}_2 \geq p^*$, it accepts the offer.

(2) Otherwise, it checks if it can reduce the SQ of other devices to be able to host the migrated note, while still improving the overall SQ. Gateway 2 calculates the optimal allocation considering if the candidate node would have connected to it. Then it calculates $\bar{u}$, which shows the utility loss if node 4 had connected to gateway 2. The offer is acceptable if the decrease in the overall utility of gateway 2 is less than the gained utility of gateway 1, i.e., $\overset{+}{u} > \bar{u}$.

If the offer is accepted, the initiator (i.e., gateway 1) is informed. Then it notifies the migration candidate (e.g., device 5). The migration candidate receives a request for connection from the new gateway (i.e., gateway 2), and it joins the new network. In this case, the current trade is terminated. The gateways update the SQ levels of their connected nodes according to the new situation.

A.1. Outcome of Migration. After the migration trade between two gateways, these three situations are possible:

(1) For both gateways, the connected devices reach their highest SQ level. It means that these two gateways have reached their local optimal configuration, and therefore no more trade is needed between them.

(2) For both gateways, the SQ level of connected devices can be improved but there are no resources left (i.e., $\hat{R}_1 \times \hat{P}_1 = 0$ and $\hat{R}_2 \times \hat{P}_2 = 0$). In this case, no more migration can help, and the gateways can only exchange to improve the overall utility.

(3) On one gateway the devices are operating at their highest SQ level, while on the other gateway there are not enough resources left to increase the SQ level of devices (e.g., $\hat{R}_2 \times \hat{P}_2 = 0$). In this case, the gateways keep migrating the devices until either the situation changes to (1) or (2) or there is no more candidate to migrate.

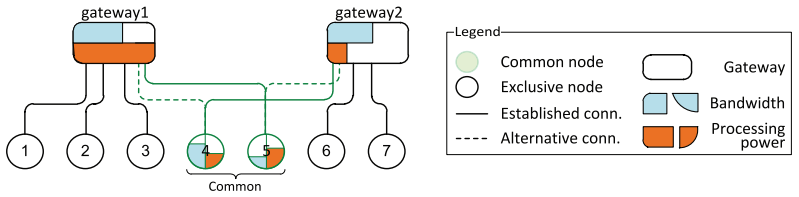


Fig. 7. An example for exchange trade.

It is worth emphasizing that in migration, the gateways will not encounter a ping-pong effect in trading (migrating nodes between two gateways back and forth). The intuitive reason is that we migrate the node only if the gained utility on the source gateway is strictly greater than the utility loss on the destination gateway. Hence, the reverse move is certainly disadvantageous and harmful.

B. Exchange

Before proceeding to the details of the exchange move, we introduce some metrics that will be used to make decisions in the trade.

Definition 4.5. Marginal value is the increase (or decrease) in utility obtained for a fixed increase (or decrease) in the resources allocated to a device (Boyd and Vandenberghe 2004). Loosely speaking, it is the derivative of the utility function (i.e., slope of lines in Figure 4).

The key policy of our trade scheme is: *Having a unit of resource available on the gateway, allocate it to the device with the highest marginal value.* In other words, we allocate more resources to the IoT device that benefits from it the most. According to the concavity property (i.e., Proposition 3), the marginal value decreases (or remains the same, finitely) as the allocated resource to the node increases.

B.1. Which Gateway and Which IoT Device? If a gateway has allocated the resources to its connected nodes such that they are all operating at their highest SQ level, this gateway has no incentive to initiate an exchange. Therefore, an exchange is initiated by a gateway that does not have enough resources left to increase the SQ of its nodes (i.e., $\hat{R}_g \times \hat{P}_g = 0$).

In the following, we present the details of the exchange trade using the example shown in Figure 7.

- **Picking the candidate node:** If the initiator gateway (e.g., gateway 1) has more than one candidate to consider for the exchange, it prioritizes the common node that uses the scarce resource the most. For instance, in our example shown in Figure 7, the scarce resource of gateway 1 is the processing power. Hence, the common node that consumes the most processing power on gateway 1 is the candidate to be exchanged. The intuition is that exchanging this node might resolve the heterogeneity issue and free up more resources for other nodes to increase their SQ level (and consequently the overall utility).
- **Offer an exchange:** Once the candidate node is selected (node 5 in our example), the gateway sends an exchange offer to the other gateway (i.e., gateway 2), which includes this information:
  - The EFC_5
  - The current configuration of device 5, i.e., (u_5^*, r_5^*, p_5^*)
  - Its remaining unused resources, $\hat{R}_1$ and $\hat{P}_1$ (at least one of them is zero)

The recipient of the offer, gateway 2, needs to first select one candidate node and then evaluate the offer to check whether accepting this offer is in the direction of increasing the overall utility of the system (i.e., is a Kaldor/Hicks improvement).

- **Picking the candidate node:** If it has more than one candidate to exchange, gateway 2 prioritizes its common node, which has an opposite resource usage profile compared to the offered node. This means that if node 5 is processing intensive, gateway 2 looks for a node that is bandwidth intensive, and vice versa. In our considered example shown in Figure 7, the candidate node that gateway 2 selects for the exchange is node 4.
- **Evaluate the offer:** Since the recipient of the offer (i.e., gateway 2) has the overall information of the trade, it can make the decision to accept or reject the offer. Let $V_d(r, p)$ denote the maximum utility that device d can reach if the allocated bandwidth and processing power for it are r and p, respectively. This value can be found easily from the piecewise-linear utility function of the device (see Figure 4).
  Then, gateway 2 needs to calculate and compare the possible change in utility of the exchanged devices:
  - First, it calculates $V_5(\hat{R}_2 + r_4^*, \hat{P}_2 + p_4^*)$, which describes the utility of node 5 in case it connects to gateway 2, replacing node 4. It basically considers the amount of resources that it would have after exchange, $\hat{R}_2$ and $\hat{P}_2$, in addition to the resources that node 4 would release (i.e., r_4^* and p_4^*).
  - Then, it similarly calculates $V_4(\hat{R}_1 + r_5^*, \hat{P}_1 + p_5^*)$.

  Finally, to evaluate the received offer, gateway 2 checks the expected change in the overall utility and decides about the offer:

$$\begin{cases} \text{accept} & V_5(\hat{R}_2 + r_4^*, \hat{P}_2 + p_4^*) + V_4(\hat{R}_1 + r_5^*, \hat{P}_1 + p_5^*) > u_5^* + u_4^* \\ \text{reject} & \text{otherwise.} \end{cases} \tag{20}$$

Once the gateways agree on a trade (exchange or migration), the IoT device will be disconnected from one gateway and connect to the other one. This process is usually very fast and takes only a few milliseconds (6ms for Bluetooth Low Energy). During this time, the IoT device can buffer its data (if any) and transmit it after connecting to the new gateway. The state of process (i.e., execution context) on the gateway is very small for the IoT applications (negligible for our ECG processing case study presented in Section 5.1). The reason is that most of the IoT applications perform repeatedly an operation on the stream of input data. This operation is performed on a short "window" of buffered data. Processing of one "window" of data is independent of previous windows. Therefore, the main communication overhead for exchange/migration is the negotiation of gateways for trading.

5 EVALUATION AND EXPERIMENTAL RESULTS

In order to evaluate the effectiveness of our proposed mechanism, we conducted experiments and case studies, which include real-world measurements on an actual IoT device and trace-driven network simulation to investigate the behavior of the system.

5.1 IoT Application Case Study

We consider a healthcare monitoring application in which the IoT devices capture the ECG signals from patients for abnormality detection and diagnosis. For our ECG analysis flow, we use actual patient ECG data records from the *MIT-BIH Arrhythmia Database* (Moody and Mark 2001). It provides signals annotated by medical experts, which are used for feature extraction and classification. Original signals were sampled at 360Hz. Down- and up-sampling have been

Table 1. Input Data Rates and Transmission Data Rates for Different SQ Levels and Offloading Levels

SQ level	Sampling freq. [Hz]	Input data rate x_d [B/s]	Transmission rate r_d [B/s] for offloading after a certain pipeline stage			
			Stage 1	Stage 2	Stage 3	Stage 4
1	180	720	720	360	104	1
2	360	1440	1440	720	192	1
3	720	2880	2880	1440	372	1
4	1440	5760	5760	2880	564	1
5	2000	8000	8000	4000	1024	1

used to generate ECG signals of differing sampling rates. These different sampling rates can be considered in two application scenarios:

(1) In a medical scenario, the specialist requires the ECG trace to have a minimum sampling rate and SQ. This is the constraint given by the medical specialist (e.g., 360Hz). Then only the SQ levels higher than this constraint would be considered (e.g., 360Hz, 720Hz, etc.). A higher sampling rate provides more accurate information, which increases the probability of correct diagnosis.
(2) In the normal usages (e.g., fitness and well-being), the user may accept a lower quality in collected data while still maintaining some crucial functionality, in return for longer battery lifetime.

We consider five SQ levels or input data rates. Our ECG analysis flow is pipelined in four stages: (1) filtering, (2) segmentation and heart beat detection, (3) feature extraction, and (4) classification and diagnosis. More details on our ECG analysis flow can be found in Samie et al. (2016c). Table 1 summarizes the parameter values for combinations of SQ levels and offloading levels stages for our ECG analysis application.

A key component of the system model is determining the utility functions of each device. The SQ value of each combination of SQ level and ECG processing stage was set proportional to the ratio of the sampling frequency of ECG signal divided by the maximum available ECG sampling frequency (2kHz) and multiplied by a diminishing factor of $(0.9)^{i-1}$, where i is the SQ level. We also enable the creation of more complex profiles of IoT devices, i.e., by allowing the user to specify a factor of how important high SQ levels are for this device.

5.2 Experimental Setup

To conduct the experiments, a combination of experimentally derived data enhanced with nominal data from data sheets of commercial devices is used for the values of our model parameters. Regarding the available energy of the IoT device (e_d), a battery consumption model of each IoT node is composed based on the instrumented CPU utilization. To complete the battery model of the IoT nodes, we choose a rechargeable lithium-ion coin cell battery with nominal capacity of up to $420mAh$ (Dittrich et al. 2012). We use a realistic discharge model for the battery using (Zhu et al. 2013) for various values of discharge currents to evaluate the available energy for Equation (4).

An ARM Cortex-M3 device is considered as the gateway (Mainetti et al. 2011). The energy consumption values were acquired by hardware measurements and profiling the execution of the ECG analysis flow for all combinations of SQ levels and processing stages to measure the values

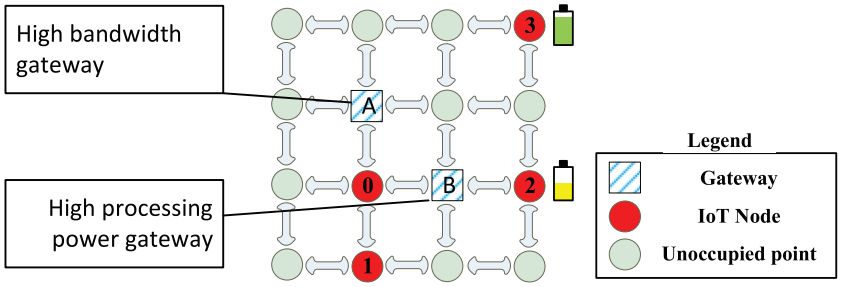


Fig. 8. An example of an examined IoT-based system.

of our parameters, e.g., $C_d(\cdot)$, $p(r_{d_{ij}})$, and so forth. The energy consumption of ECG acquisition was calculated based on (Hann 2013).

Bluetooth Low Energy (BLE) is used for communication between IoT devices and the gateway. The power consumption value of data transmission is $0.153\mu W$ based on (Smith 2011) and transmission latency is $4\mu s/bit$ (Siekkinen et al. 2012). Since BLE exploits adaptive frequency hopping mechanisms, the probability of interference is very low and even if it happens, it would be for a very short period of time. Moreover, the gateways communicate with each other using either a different medium (wired) or a dedicated frequency band such as sub-1GHz (different from the band that IoT devices are using to transmit their data). Therefore, the interference among gateways is avoided and would not affect the IoT devices and their battery.

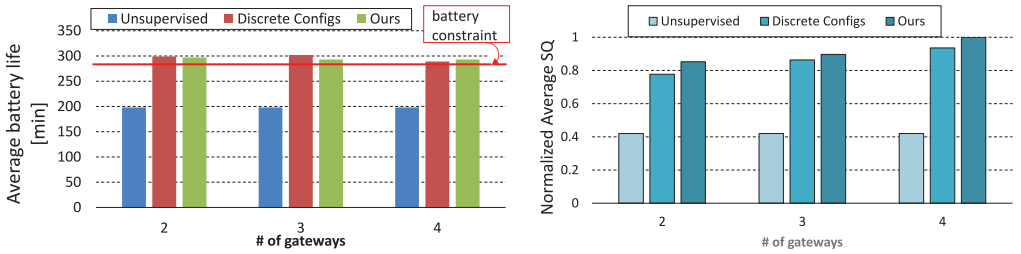
5.3 Simulation Framework

Our following experimental analysis has been conducted using a simulator, created as a part of this work, which implements all the models of the different devices of the system. The inherent distributed nature of the system under analysis led us to create a simulator where each one of the previously described agents (gateways and IoT nodes) corresponds to a different processing entity. In this way, message exchange and execution of our proposed resource allocation scheme can take place in parallel, thus creating a realistic interplay of the involved agents.

A very important goal was to be able to capture the topological characteristics of an examined IoT-based system. In this respect, our simulation evolves in a virtual grid of devices, where in each position of the grid there can be either an IoT node or a gateway. Using this feature, we simulate real-life conditions where communication of different nodes is affected by their distance and interference with the rest of working devices in proximity. Every node is unique and can have different initial values in its characteristics, such as battery capacity for an IoT node or available bandwidth and processing capabilities in a gateway.

An example of an instance of the topologies that can be mapped to our simulation framework is illustrated in Figure 8. In this case, there is a 4x4 grid with two gateways and four IoT nodes. The rest of the grid points are unoccupied but are available for use in other scenarios. Each one of the devices has its own characteristics; e.g. IoT node 3 has increased battery capacity compared to IoT node 2, whose battery is low. The same applies for gateways, where A is rich in bandwidth while B is rich in processing power.

From the point of view of the implementation, the simulator has been created as a C program running on Linux OS. Each different entity is mapped to a different process of the OS, in order to provide encapsulation and enable parallel execution. Interprocess, i.e., internode, communication is achieved by writing data packets in the shared memory of each entity. These data packets correspond to either signals or data that are transmitted from one node to the other. Semaphores



(a) Average battery lifetime (b) Normalized average SQ level

Fig. 9. The scenario of low number of IoT devices.

have also been incorporated in the communication scheme, in order to help its orchestration and protect against the violation of the memory space of a process by another one.

5.4 Results

We consider an IoT system that is located inside a big hospital room or a clinic ward. We assume a number of patients whose vital signals and mainly ECG signals are monitored by wearable IoT edge devices. The room is simulated by an 8x8 grid where a different number of gateways and IoT edge devices are present per scenario.

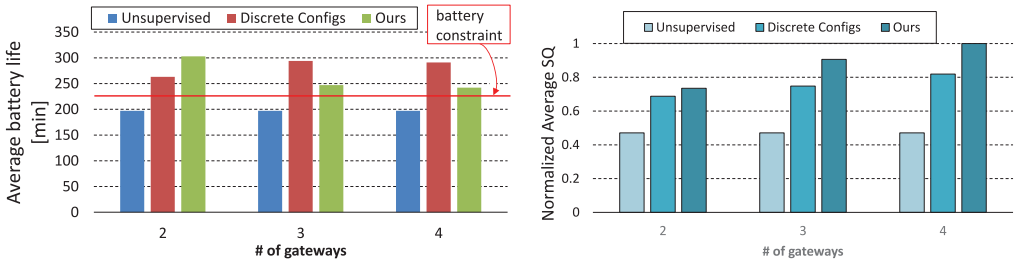
We identify three distinct scenarios where the number of patients inside the room is (1) low, (2) medium, or (3) high. This affects the requirements of the connection to the available gateways, leading to increased demand as the number of patients rises. Furthermore, for each scenario we examine a different number of available gateways. The gateways are located diagonally in the room as shown in Figure 8.

The characteristics of each IoT node also differ since they are mapped to different patients. Each device has its own initial battery capacity and expected battery lifetime, accompanied by a coefficient indicating the importance of increased SQ level to the specific user.

The last critical design alternative examined in each scenario is the algorithm employed by the gateways in order to accomplish the binding and allocation for the operation points of each IoT node. The available options are to execute the resource allocation algorithm either with discrete configuration points (Samie et al. 2016c) or with our proposed scheme. In both cases, the ability for gateways to migrate or exchange nodes is activated. Another comparison is with the scheme where devices operate in the absence of any resource management scheme, where the IoT node operates in a medium SQ level, and only communicate their processing outcome to the gateway.

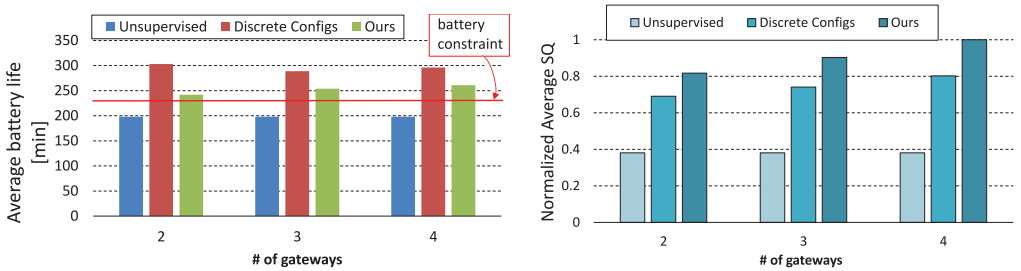
The results of the three examined scenarios are presented with two metrics: (1) average battery lifetime of the IoT nodes and (2) normalized average achieved SQ level. The first examined scenario was the one with a low number of patients (i.e., IoT edge devices). Figure 9(a) illustrates the average achieved battery lifetime of the devices. This metric is chosen since there is no global expected lifetime values for all devices but each one operates under a unique constraint. To visualize the expected battery lifetime, the red line on the figure denotes the average expected battery lifetime, taking into account all devices involved in the scenario. In both techniques for SQ management, all devices meet their battery lifetime constraint. On the contrary, the unsupervised system behaves very poorly and misses the majority of the constraints.

In Figure 9(b), the average achieved SQ level is presented, normalized in respect to the highest achieved value in this scenario. We can see that the proposed resource allocation scheme operating on a continuous utility function achieves better SQ levels compared to the scheme with a discrete utility function in all cases. Since the number of devices is low, the gain remains low as in both



(a) Average battery lifetime (b) Normalized average SQ level

Fig. 10. The scenario of medium number of IoT nodes.



(a) Average battery lifetime (b) Normalized average SQ level

Fig. 11. The scenario of high number of IoT nodes.

schemes there are enough available resources. The gain in SQ level of the proposed scheme comes at the price of decreased average battery lifetime of IoT nodes compared to the discrete version. However, this is acceptable since the battery lifetime constraints are still met.

This presented system behavior remains the same throughout the examined scenarios with a medium and high number of devices as illustrated in Figure 10 and Figure 11. As expected in these cases, as the number of IoT devices increases, the available resources on the gateways become more and more scarce. Consequently, the ability of the proposed solution to fully utilize available bandwidth and processing resources of the gateway results in improvement in achieved SQ levels. The same overall SQ level cannot be achieved by the discrete counterpart of the resource allocation scheme, since it suffers from the fragmented utilization of its resources. These gains reach up to 22% and 24.6% in the cases of medium and high number of edge devices, respectively. Overall, the gains compared to the unsupervised version of the system start from 56% and can reach up to more than 100%.

Figure 12 presents the overhead of the resource management negotiations on the rest of the system communication, which corresponds to the data traffic generated by the applications running on the IoT nodes. Assuming that Tot_Mngmt_Msgs is the total number of messages exchanged for resource negotiations and Tot_App_Msgs is the total number of application-related messages sent by the IoT nodes to gateways, then the overhead in message count is defined as

$$\frac{Tot\_Mngmt\_Msgs}{Tot\_App\_Msgs} \times 100\%, \tag{21}$$

while the overhead in transmitted data volume equals

$$\frac{Sizeof(Tot\_Mngmt\_Msgs)}{Sizeof(Tot\_App\_Msgs)} \times 100\%. \tag{22}$$

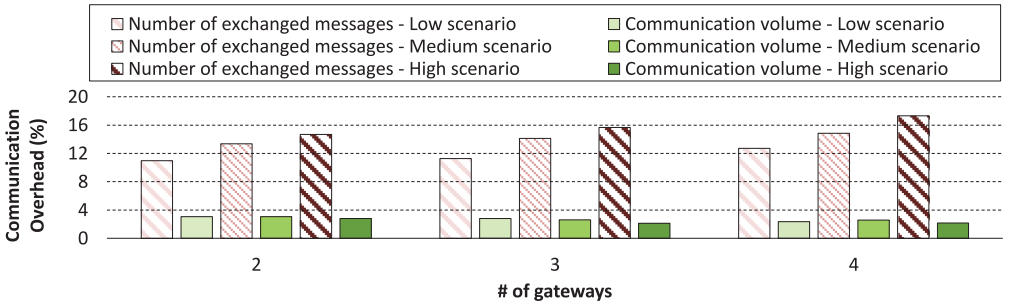


Fig. 12. Communication overhead.

Experiments showed that the overhead in the number of exchanged messages ranges from 10% to 18% and is higher for increased number of IoT nodes (patients) and gateways, since more devices need to communicate in order to designate the operating points of the system. Conversely, the overhead on communication volume is almost constant and lower than 3% in all cases. To interpret this result, the differences in the structure of resource management and application-related messages should be taken into account. Resource negotiation messages are small and contain only a few values as described in Section 4.7, while application messages transmitted to the gateway can contain many values according to the SQ and off-loading level of each device, e.g., a complete heartbeat window sent for processing. As a result, the resource negotiation-related messages constitute a very small fraction of the total communication volume.

6 CONCLUSIONS

In this article, we studied the joint problem of binding and resource allocation for multigateway IoT systems, where the IoT devices can provide different service quality levels and can offload a share of their workload. The SQ management has to fulfill constraints for the battery lifetime of IoT devices, communication bandwidth to the gateway, and processing capability of the gateway (for offloading). We present an integer programming formulation for this problem and decompose it into separate device and gateway problems. We showed that the gateway problem is NP-hard and practically impossible to be solved in a central fashion. We proposed a distributed agent-based mechanism to find the suboptimal solutions. We demonstrate the effectiveness of our proposed approach using a case study of ECG processing in a personal healthcare monitoring application. The experiments show that our solution achieves up to 56% accumulated SQ compared to an unsupervised system and up to 24.6% compared to a resource allocation scheme with discrete utility configurations, while they both meet the constraints of edge devices.

REFERENCES

Arif Ahmed and Ejaz Ahmed. 2016. A survey on mobile edge computing. In *International Conference on Intelligent Systems and Control (ISCO'16)*.

Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. 2015. Internet of Things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials* 17, 4 (2015), 2347–2376.

Julio Barbancho, Carlos Leon, Javier Molina, and Antonio Barbancho. 2006. Giving neurons to sensors. QoS management in wireless sensors networks. In *IEEE Conference on Emerging Technologies and Factory Automation*. IEEE, 594–597.

Sangeeta Bhattacharya, Abusayeed Saifullah, Chenyang Lu, and Gruia-Catalin Roman. 2010. Multi-application deployment in shared sensor networks based on quality of monitoring. In *16th IEEE Real-Time and Embedded Technology and Applications Symposium*. 259–268.

Daniele Bortolotti, Mauro Mangia, Andrea Bartolini, Riccardo Rovatti, Gianluca Setti, and Luca Benini. 2016. Energy-aware bio-signal compressed sensing reconstruction on the WBSN-gateway. *IEEE Transactions on Emerging Topics in Computing* (2016).

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

Rubén Braojos, Ivan Beretta, Jeremy Constantin, Andreas Burg, and David Atienza. 2014. A wireless body sensor network for activity monitoring with low transmission overhead. In *IEEE International Conference on Embedded and Ubiquitous Computing (EUC'14)*. 265–272.

Luca Catarinucci, Danilo De Donno, Luca Mainetti, Luca Palano, Luigi Patrono, Maria Laura Stefanizzi, and Luciano Tarricone. 2015. An iot-aware architecture for smart healthcare systems. *IEEE Internet of Things Journal* 2, 6 (2015), 515–526.

Dazhi Chen and Pramod K. Varshney. 2004. QoS support in wireless sensor networks: A survey. In *International Conference on Wireless Networks*, Vol. 233. 1–7.

Lei Chen and Wendi B. Heinzelman. 2005. QoS-aware routing based on bandwidth estimation for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications* 23, 3 (2005), 561–572.

Xu Chen. 2015. Decentralized computation offloading game for mobile cloud computing. *IEEE Transactions on Parallel and Distributed Systems* 26, 4 (2015), 974–983.

Shao-Yi Chien, Wei-Kai Chan, Yu-Hsiang Tseng, Chia-Han Lee, V. Srinivasa Somayazulu, and Yen-Kuang Chen. 2015. Distributed computing in IoT: System-on-a-chip for smart cameras as an example. In *Asia and South Pacific Design Automation Conference (ASP-DAC'15)*. 130–135.

Sung-woo Cho and Ashish Goel. 2006. Pricing for fairness: Distributed resource allocation for multiple objectives. In *ACM Symposium on Theory of Computing*. 197–204.

Delphine Christin, Andreas Reinhardt, Parag Mogre, and Ralf Steinmetz. 2009. Wireless sensor networks and the internet of things: Selected challenges. In *Proceedings of the 8th GI/ITG KuVS Fachgespräch Drahtlose Sensornetze*.

Jules L. Coleman. 1979. Efficiency, utility, and wealth maximization. *Hofstra Law Review* 8 (1979), 509.

Thomas Dittrich, Chen Menachem, Y. Herzel, and A. Lou. 2012. *Lithium Batteries for Wireless Sensor Networks*. Technical Report. Tadiran Batteries.

Niroshinie Fernando, Seng W. Loke, and Wenny Rahayu. 2013. Mobile cloud computing: A survey. *Future Generation Computer Systems* 29, 1 (2013), 84–106.

Ashish Goel and Hamid Nazerzadeh. 2014. Price-based protocols for fair resource allocation: Convergence time analysis and extension to Leontief utilities. *ACM Transactions on Algorithms (TALG)* 10, 2 (2014), Article 5.

Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems* 29, 7 (2013), 1645–1660.

Matthew W. Hann. June 2013. Ultra Low Power, 18 bit Precision ECG Data Acquisition System.

Moeen Hassanalieragh, Alex Page, Tolga Soyata, Gaurav Sharma, Mehmet Aktas, Gonzalo Mateos, Burak Kantarci, and Silvana Andreescu. 2015. Health monitoring and management using Internet-of-Things (IoT) sensing with cloud-based processing: Opportunities and challenges. In *IEEE International Conference on Services Computing (SCC'15)*. 285–292.

Weiwei He, Shuang-Hua Yang, Lili Yang, and Ping Li. 2015. In-network data processing architecture for energy efficient wireless sensor networks. In *IEEE World Forum on Internet of Things (WF-IoT'15)*.

Wendi B. Heinzelman, Amy L. Murphy, Hervaldo S. Carvalho, and Mark A. Perillo. 2004. Middleware to support sensor network applications. *IEEE Network* 18, 1 (2004), 6–14.

Jörg Henkel, Santiago Pagani, Hussam Amrouch, Lars Bauer, and Farzad Samie. 2017. Ultra-low power and dependability for IoT devices (Invited paper for IoT technologies). In *Design, Automation & Test in Europe Conference & Exhibition (DATE'17)*. 954–959.

Dong Huang, Ping Wang, and Dusit Niyato. 2012. A dynamic offloading algorithm for mobile computing. *IEEE Transactions on Wireless Communications* 11, 6 (2012), 1991–1995.

Sungwook Kim. 2015. Nested game-based computation offloading scheme for mobile cloud IoT systems. *Journal on Wireless Communications and Networking* 2015, 1 (2015), 229–239.

Andreas Kliem and Odej Kao. 2015. The Internet of Things resource management challenge. In *IEEE International Conference on Data Science and Data Intensive Systems*. 483–490.

Dejan Kovachev, Tian Yu, and Ralf Klamma. 2012. Adaptive computation offloading from mobile devices into the cloud. In *International Symposium on Parallel and Distributed Processing with Applications (ISPA'12)*. 784–791.

Karthik Kumar, Jibang Liu, Yung-Hsiang Lu, and Bharat Bhargava. 2013. A survey of computation offloading for mobile systems. *Mobile Networks and Applications* 18, 1 (2013), 129–140.

Jeongho Kwak, Yeongjin Kim, Joohyun Lee, and Song Chong. 2015. DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems. *IEEE Journal on Selected Areas in Communications* 33, 12 (2015), 2510–2523.

Mihai T. Lazarescu. 2014. Internet of Things low-cost long-term environmental monitoring with reusable wireless sensor network platform. In *Internet of Things*. 169–196.

In Lee and Kyoochun Lee. 2015. The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Business Horizons* 58, 4 (2015), 431–440.

SangJoon Lee, Jungkuk Kim, and Myoungho Lee. 2011. A real-time ECG data compression and transmission algorithm for an e-health device. *IEEE Transactions on Biomedical Engineering* 58, 9 (2011), 2448–2455.

Ling Li, Shancang Li, and Shanshan Zhao. 2014. QoS-aware scheduling of services-oriented Internet of Things. *IEEE Transactions on Industrial Informatics* 10, 2 (2014), 1497–1505.

Zhiyuan Li, Cheng Wang, and Rong Xu. 2001. Computation offloading to save energy on handheld devices: A partition scheme. In *International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES'01)*. 238–246.

Gregorio López, Víctor Custodio, and José Ignacio Moreno. 2010. LOBIN: E-textile and wireless-sensor-network-based platform for healthcare monitoring in future hospital environments. *IEEE Transactions on Information Technology in Biomedicine* 14, 6 (2010), 1446–1458.

Qicheng Ma, David C. Parkes, and Matthew D. Welsh. 2007. A utility-based approach to bandwidth allocation and link scheduling in wireless networks. In *International Workshop on Agent Technology for Sensor Networks (ATSN'07)*.

Luca Mainetti, Luigi Patrono, and Antonio Vilei. 2011. Evolution of wireless sensor networks towards the Internet of Things: A survey. In *International Conference on Software, Telecommunications and Computer Networks (SoftCOM'11)*. 1–6.

Yuyi Mao and Jun Zhang. 2016. Dynamic computation offloading for mobile-edge computing with energy harvesting devices. *IEEE Journal of Solid-State Circuits* 51, 3 (2016), 712–723.

Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. 2012. Internet of Things: Vision, applications and research challenges. *Ad Hoc Networks* 10, 7 (2012), 1497–1516.

George B. Moody and Roger G. Mark. 2001. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine* 20, 3 (2001), 45–50.

Amy Murphy and Wendi Heinzelman. 2003. *Milan: Middleware Linking Applications and Networks*. Technical Report.

Guido Oddi, Antonio Pietrabissa, Francesco Delli Priscoli, Francisco Facchinei, Laura Palagi, and Andrea Lanna. 2015. A QoE-aware dynamic bandwidth allocation algorithm based on game theory. In *Mediterranean Conference on Control and Automation (MED'15)*. 979–985.

Chris Raphael. 2016. Why Edge Computing Is Crucial for the IoT. Retrieved from http://www.rtinsights.com/why-edge-computing-and-analytics-is-crucial-for-the-iot/.

Laurynas Riliskis, James Hong, and Philip Levis. 2015. Ravel: Programming IoT applications as distributed models, views, and controllers. In *International Workshop on Internet of Things towards Applications*.

Ola Salman, Imad Elhajj, Ayman Kayssi, and Ali Chehab. 2015. Edge computing enabling the Internet of Things. In *IEEE World Forum on Internet of Things (WF-IoT'15)*. 603–608.

Farzad Samie, Lars Bauer, and Jörg Henkel. 2016a. IoT technologies for embedded computing: A survey. In *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'16)*.

Farzad Samie, Vasileios Tsoutsouras, Sotirios Xydis, Lars Bauer, Dimitrios Soudris, and Jörg Henkel. 2016b. Computation offloading management and resource allocation for low-power iot edge devices. In *IEEE 3rd World Forum on Internet of Things (WF-IoT'16)*.

Farzad Samie, Vasileios Tsoutsouras, Sotirios Xydis, Lars Bauer, Dimitrios Soudris, and Jörg Henkel. 2016c. Distributed QoS management for Internet of Things under resource constraints. In *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'16)*. IEEE Press.

Stefania Sardellitti, Gesualdo Scutari, and Sergio Barbarossa. 2015. Joint optimization of radio and computational resources for multicell mobile-edge computing. *IEEE Transactions on Signal and Information Processing over Networks* 1, 2 (2015), 89–103.

Zhengguo Sheng, Chinmaya Mahapatra, Victor C. M. Leung, Min Chen, and Pratap Kumar Sahu. 2015. Energy efficient cooperative computing in mobile wireless sensor networks. *IEEE Transactions on Cloud Computing* 6, 1 (2015), 114–126.

Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3, 5 (2016), 637–646.

Matti Siekkinen, Markus Hiienkari, Jukka K. Nurminen, and Johanna Nieminen. 2012. How low energy is Bluetooth low energy? comparative measurements with ZigBee/802.15.4. In *IEEE Wireless Communications and Networking Conference Workshops (WCNCW'12)*. 232–237.

Phil Smith. 2011. Comparing low-power wireless technologies. *Tech Zone, Digikey Online Magazine, Digi-Key Corporation*.

Brent A. Smolinski. 2000. Approximating the 0-1 multiple knapsack problem with agent decomposition and market negotiation. In *Intelligent Problem Solving. Methodologies and Approaches*. Springer, 296–306.

Kevin Townsend, Carles Cufí, Chris Wang, and Robert Davidson. 2014. *Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking*. O'Reilly Media.

Yating Wang, Ray Chen, and Ding-Chau Wang. 2015. A survey of mobile cloud computing applications: Perspectives and challenges. *Wireless Personal Communications* 80, 4 (2015), 1607–1623.

Roy Want, Bill N. Schilit, and Scott Jenson. 2015. Enabling the Internet of Things. *IEEE Computer* 48, 1 (2015), 28–35.

Geng Wu, Shilpa Talwar, Kerstin Johnsson, Nageen Himayat, and Kevin D. Johnson. 2011. M2M: From mobile to embedded Internet. *IEEE Communications Magazine* 49, 4 (2011), 36–43.

Feng Xia. 2008. QoS challenges and opportunities in wireless sensor/actuator networks. *Sensors* 8, 2 (2008), 1099–1110.

Feng Xia, Wenhong Zhao, Youxian Sun, and Yu-Chu Tian. 2007. Fuzzy logic control based QoS management in wireless sensor/actuator networks. *Sensors* 7, 12 (2007), 3179–3191.

Changjiu Xian, Yung-Hsiang Lu, and Zhiyuan Li. 2007. Adaptive computation offloading for energy conservation on battery-powered systems. In *International Conference on Parallel and Distributed Systems*, Vol. 2. 1–8.

Thomas Zachariah, Noah Klugman, Bradford Campbell, Joshua Adkins, Neal Jackson, and Prabal Dutta. 2015. The Internet of Things has a gateway problem. In *Mobile Computing Systems and Applications (HotMobile'15)*. 27–32.

Ben Zhang, Nitesh Mor, John Kolb, Douglas S. Chan, Nikhil Goyal, Ken Lutz, Eric Allman, John Wawrzynek, Edward Lee, and John Kubiatowicz. 2015. The cloud is not enough: Saving IoT from the cloud. In *7th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud'15)*. 21–21.

Cong Zhu, Xinghu Li, Lingjun Song, and Liming Xiang. 2013. Development of a theoretically based thermal model for lithium ion battery pack. *Journal of Power Sources* 223 (2013), 155–164.

Qian Zhu, Ruicong Wang, Qi Chen, Yan Liu, and Weijun Qin. 2010. IoT gateway: Bridging wireless sensor networks into Internet of Things. In *Embedded and Ubiquitous Computing (EUC'10)*. 347–352.

Received August 2016; revised April 2017; accepted August 2017