IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

Received 16 March 2014; revised 11 October 2014; accepted 19 October 2014. Date of publication 22 October 2014;
date of current version 4 February 2015.

Digital Object Identifier 10.1109/TETC.2014.2364915

Toward QoI and Energy-Efficiency in Internet-of-Things Sensory Environments

CHI HAROLD LIU[1], (Member, IEEE), JUN FAN[1], JOEL W. BRANCH[2], (Member, IEEE), AND KIN K. LEUNG[3], (Fellow, IEEE)

[1]School of Software, Beijing Institute of Technology, Beijing 100081, China
[2]IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA
[3]Department of Electrical and Electronic Engineering and Computing, Imperial College London, London SW7 2AZ, U.K.
CORRESPONDING AUTHOR: C. H. LIU (liuchi02@gmail.com)

This work was supported by the National Natural Science Foundation of China under Grant 61300179.

ABSTRACT Considering physical sensors with certain sensing capabilities in an Internet-of-Things (IoTs) sensory environment, in this paper, we propose an efficient energy management framework to control the duty cycles of these sensors under quality-of-information (QoI) expectations in a multitask-oriented environment. Contrary to past research efforts, our proposal is transparent and compatible both with the underlying low-layer protocols and diverse applications, and preserving energy-efficiency in the long run without sacrificing the QoI levels attained. In particular, we first introduce the novel concept of QoI-aware sensor-to-task relevancy to explicitly consider the sensing capabilities offered by a sensor to the IoT sensory environments, and QoI requirements required by a task. Second, we propose a novel concept of the critical covering set of any given task in selecting the sensors to service a task over time. Third, energy management decision is made dynamically at runtime, to reach the optimum for long-term application arrivals and departures under the constraint of their service delay. We show a case study to utilize sensors to perform environmental monitoring with a complete set of performance analysis. We further consider the signal propagation and processing latency into the proposal, and provide a thorough analysis on its impact on average measured delay probability.

INDEX TERMS Internet-of-Things, energy management, quality-of-information.

I. INTRODUCTION

The Internet of Things (IoT, [1], [2]) (including cyber physical systems [3]) represents an evolution in computerized interconnectivity. Not only traditional computers, but also smart autonomous devices will seamlessly interconnect via machine-to-machine (M2M) technologies [4]–[6] to provide and consume new classes of service such as intelligent shopping, smart product management and home automation. It is expected that such devices will largely include physical sensors and RFID tags embedded in various hosts (e.g., vehicles, buildings, habitats, humans, utility grids, containers, garments, smart phones, etc.), enabling constant analysis of their environmental and operational states [7], [8]. Examples include real-time supply chain monitoring, remote patient monitoring, and infrastructures monitoring [9]. Further goals include using such capabilities to enable highly intelligent and responsive actuation, for example through dynamic public transit scheduling or efficient and predictive utility (electricity, water) management.

Fig. 1 shows some architectural elements representative of an IoT system. They include a collection of applications and services at the top layer that act on the collective knowledge gathered by the computerized intelligence embedded in sensors at the bottom layer. As the figure shows, these sensors may belong to various domains (e.g., different municipal agencies), which can be all accessed by an individual IoT application (e.g., representing a wide-area traffic management or emergency response service). Between these two layers reside a number of supporting middleware services that facilitate the interaction between the applications and sensors. Such distributed services include the following: (1) an application gateway (*IoT AppGW* in the figure)

2168-6750 © 2014 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

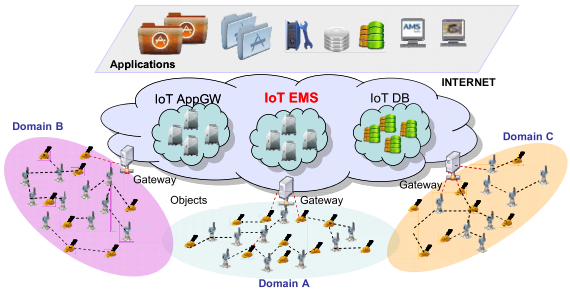

FIGURE 1. The overall IoT architecture, where a middleware layer (composed of the IoT application gateway, EMS, and real-time operational database) bridges a variety of applications with the physical sensors.

for information collection, processing, and delivery; (2) a database (*IoT DB*) for information archival and query; and (3) an energy management server (*IoT EMS*) for adjusting the trade-off between devices' resource consumption and provided data quality, which is especially important for energy limited devices. While all the components in Fig. 1 require unique advancements to manifest the IoT vision, we focus our work on the challenge of balancing sensor nodes' energy and data quality management.

There are various technical challenges in sensor energy and data quality management. A major one that drives our work involves the large-scale management of heterogeneous devices that are expected to populate IoT systems. A great many sensor types, manufacturers, protocols, etc. are expected to co-exist here and hence, any solution must be designed to operate as expected regardless of the device configuration. Regarding energy management, this motivates the need for a universal management approach that attempts to control MAC (medium access control) level energy consumption of nodes, as motivated by previous research [7], [10]. Furthermore, an efficient management scheme should minimize the transmission of control messages crossing different domains, and thus we are seeking a *long-term* optimal solution. In regards to data quality management, a universal measure of expression can be found in recent work in quality-of-information (QoI) management. Broadly speaking, QoI relates to the ability to judge whether information is *fit-for-use* for a particular purpose [11]–[13]. For the purposes of this paper, we will assume that QoI is characterized by a number of attributes including accuracy, latency, and physical context (specifically, sensor coverage in this paper [11]).

To address the aforementioned challenges, we aim to design an energy management service (and supporting algorithms) that is transparent to and compatible with any lower layer protocols and over-arching applications, while providing *long-term* energy-efficiency under the satisfactory QoI constraints. In support of our design, we first introduce the new concept of "sensor-to-task relevancy" to explicitly consider the sensing capabilities offered by a sensor (or a set thereof) to the applications and QoI requirements required by a task. Second, we use the generic information fusion function to compute the "critical covering set" of any given task in selecting the sensors to service a task over time. Third, we propose a runtime energy management framework based on the previous design elements to control the duty cycles of a sensor in the long run, i.e., the control decision is made optimally considering the long-term task usage statistics where the service delay of each task serves as the constraint. Then, an extensive case study related to environmental monitoring is given to demonstrate the ideas and algorithms proposed in this paper, and a simulation is made to support all performance analysis. Lastly, we further consider the signal propagation and processing latency into our previous proposal to both theoretically and experimentally investigate its impact on the measured delay probability. Finally, we provide some potential implementation guidelines to make the energy management framework more applicable under realistic scenarios. It should be noted that this is first-of-a-kind research that manages the energy usage of a variety of sensors from different domains, irrespective of how the provided sensing capabilities will be used by different applications.

The rest of this paper is organized as follows. Section II presents related research efforts. The system model, including the system flow of the proposed efficient energy management framework, is described in Section III. The sensor-to-task relevancy and the critical covering set are introduced in Section IV, and the optimization problem of efficient energy management is formulated in Section V, where several solutions are also given and analyzed. In Section VI the modeling of signaling overhead is thoroughly discussed and analyzed. In Section VII, a case study of environmental monitoring is explained in detail, and its simulation results are presented. Implementation guidelines are given in Section VIII. Finally, concluding remarks are drawn in Section IX. This paper largely extends [14], by introducing the delay model for all tasks in a probabilistic manner (see Section V-B), giving the explicit definition of the weight factor in the duty cycle optimization (see Section V-C), demonstrating extensive performance evaluation results (see Section VII), and adding the modeling and analysis of signal propagation and processing latency (see Section VI).

A summary of important symbols used in this paper is listed in Table 1.

II. RELATED WORK

For the definition of sensor utility, [15] proposes a decentralized approach towards the design of such networks based on utility functions. [16] overviews the information-driven approach to sensor collaboration in ad hoc sensor networks, and a definition of information utility and its approximate measures are introduced. The area of QoI has been proposed recently to judge how retrieved information is fit-for-use for a particular task [17], [18]. Work described in [19] made further contributions to this area by proposing a QoI satisfaction index to describe the degree of QoI satisfaction that tasks receive from a wireless sensor network (WSN).

TABLE 1. Summary of important symbols.

Symbol	Definition (Section where the symbol is first used)
$\underline{c}_n$	sensing capability of sensor n (III-A)
E_n	initial energy reserve of sensor n (III-A)
$\bar{E}_n(t)$	residual energy of sensor n at time t (III-A)
P_n^{on}	power consumption level of sensor n when it is ON (III-A)
$\underline{q}_m$	desired QoI of task m (III-B)
L	frame length (III-C)
r_{nm}	relevancy of sensor n to task m (IV)
$f(\cdot)$	sensor-to-task relevancy function (IV)
$g_{\mathcal{S}}(\cdot)$	information fusion function (IV-A)
T	lifetime of the IoT sensory environment (V-A)
$\eta_n(t)$	runtime generalized duty cycle of sensor n up to t (V-A)
$N_n(t)$	number of switches sensor n makes up to time t (V-A)
$\Sigma_n(t)$	aggregation of the ON times of sensor n up to time t (V-A)
P_n^{sw}	energy consumed each time sensor n switches mode (V-A)
$d_{m,i}$	service delay of task m for its i-th instance (V-B)
π_m	steady state probability of task m (V-B)
$\zeta_{m,i}$	delay failure indicator of task m for its i-th instance (V-B)
ζ_m	average measured delay failure of task m (V-B)
ξ_m	delay failure threshold of task m (V-B)
τ_m	delay tolerance of task m (V-B)
β_n	weight factor in duty cycle minimization (V-C)
$\mathbf{P}$	task transition matrix for the task model (V-D)
$\Delta_n(t)$	energy consumption of sensor n from t t0 $t+L$ (V-D)
$P_m(t)$	probability that an instance of task m starts at t (V-D)
$\Psi_m(t)$	probability of at least one CCS of task m exists at t (V-D)
$\Phi_{m,i}(t)$	probability that task m's i-th instance starts at t (V-D)
$Q(\cdot)$	tail probability of the standard normal distribution (V-D)
$Z_m(t)$	fused sensory information for task m at time t (VII-A)
$W_n(t)$	retrieved information from sensor n at time t (VII-A)
δ_m, ϵ_m	accuracy requirement of task m (VII-A)
μ	parameter for exponential task instance duration (VII-A)
$l_{\min}$	minimum task instance duration (VII-A)
ω	number of delayed frames (VI-A)
l_k	the duration of the k-th instance of all tasks (VI-A)

[19] additionally describes a QoI network capacity metric to describe the marginal ability of the WSN to support the QoI requirement of a new task upon its arrival to the network. Based on these, the authors proposed an adaptive admission control scheme to optimally accommodate the QoI requirements of all incoming tasks by treating the whole WSN as a ''black box.''

Existing work describes many different schemes for WSN node scheduling [20]–[24]. In [20], Ma *et.al.* propose centralized and distributed algorithms to assign sensors with consecutive time slots to reduce the frequency of operational state transitions. In [21], the authors describe an energy-efficient scheduling scheme for low-data-rate WSNs, where both the sensors' energy requirements for specific operational states and the state transitions are considered. In [22], the authors propose a cross-layer framework to optimize global power consumption and balance the load in sensor networks. In [23], the authors utilize control theory to balance energy consumption and packet delivery success. In [24], the authors present novel schemes to reduce sleep latency, while achieving balanced energy consumption among sensor nodes with energy harvesting capability. The authors in [25] study the QoI performance of WSN tracking applications with energy constraints, focusing on a duty-cycled network with random wake-up schedules for different nodes. In comparison, our work is different in that we explicitly consider the multi-dimensional QoI requirements of tasks and capabilities of sensors, and use this in addition to energy to determine node duty cycle schedules. This approach completely decouples relations between sensors and applications, but dynamically controls the energy consumption state of each sensor to achieve desired QoI over the long run.

Finally, we note that there has been plenty of work on MAC layer protocol designs for WSNs focusing on minimizing energy consumption in order to achieve prolonged system lifetime, such as S-MAC [26] and T-MAC [27]. In contrast to this work, our proposal is a system-level management operation and not a communications protocol; and more importantly, this proposal can work with systems that engage to MAC level energy conservation as well.

III. SYSTEM MODEL

In this section, we present the modeling of sensors, tasks and overall system architecture and flow.

A. SENSORS

We consider an IoT sensory environment that comprises a collection $\mathcal{N}$ of N sensors (indexed by $n \in \{1, 2, \ldots, N\}$), plus a gateway (the sink). The sensors form a multi-hop network to transmit data to the gateway. Each sensor is associated with certain sensing, processing and communication capabilities. The sensing capability of a sensor represents its ability to offer a certain level of QoI to a task, but independently of any specific task. The sensing capability of sensor n is described by the K-vector $\underline{c}_n \in \mathbb{R}^K$, whose entries include QoI attributes such as the measurement errors, latency in reporting, its coverage, etc. For each sensor n, the initial energy reserve is denoted as E_n, and the residual energy at time t is denoted as $\bar{E}_n(t)$.

We assume that there are only two power consumption levels for a sensor: (a) P_n^{on} when in active mode, and (b) negligibly small (relative to P_n^{on}) approximated with 0 when in sleeping mode.

B. TASKS

We consider a collection $\mathcal{M}$ of M tasks (indexed by $m \in \{1, 2, \ldots, M\}$). Each task represents a specific class of activities that may share a common spatial property but not temporal properties, such as starting time or duration. An *instance* of a task represents a single continuous period that the task is in service. For example, ''monitor the water quality at certain location'' may represent one of the M tasks, while doing so between times t_1 and t_2 or t_3 and t_4, represents two instances of the same sensing task executed over two different time periods. Each task's desired QoI is described by a K-vector $\underline{q}_m$, describing the desired accuracy, latency, coverage, etc. Note that the elements in $\underline{q}_m$ can be vectors as well, as a QoI requirement can be defined by more than one parameters, as illustrated in a case study in Section VII-A.

Finally we consider the granularity for all tasks that cannot be separated into sub-tasks. If a submitted task includes a combination of different tasks, we consider the joint set of their QoI requirements into our framework.

C. SYSTEM FLOW

We assume that our energy management scheme runs within the IoT EMS, interacting with both the applications and gateways of different underlying network domains, so that control signals can be computed, generated, and sent to the sensors (see Fig. 1).

We consider a discrete (or slotted) time system operation. Duty-cycling decisions are made by the EMS every L time slots, which define the duration of a L-slot *frame* in the system. Then decisions are sent to the gateways that coordinate control operations of the corresponding sensors. For simplicity, we assume a sensor stays active in a frame after it is woken up. Contrary to the collection $\mathcal{N}$, the gateway is assumed to have sufficient processing power and energy capacity. We assume that the frame length L is far less than the average service time of tasks and the average idle time between two consecutive instances of a task, which ensures that the probability of any task changing its status during a frame is negligible. We also assume that the current service time is known to the IoT EMS and gateway after it starts.

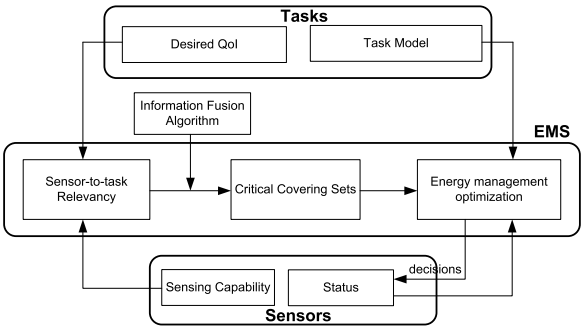

FIGURE 2. System flow of the proposed energy management framework.

Fig. 2 illustrates the procedure for the proposed energy management framework during one frame, which can be summarized as follows:

1) At the sensor deployment stage, compute the *critical covering sets* (CCSs) of each task with information fusion algorithms, based upon the sensor capabilities and the desired QoI of the tasks (see Section IV).
2) At the beginning of a frame, the EMS makes a decision on *when* to activate/deactivate which sensor and for *how long* in the current frame based on the task model and sensor status, and sends the control message back to each sensor through its gateway (see Section V).
3) During a frame, each sensor follows its predetermined waking-up schedule without further communications with the gateway until the next frame.

IV. QoI-AWARE SENSOR-TO-TASK RELEVANCY AND CRITICAL COVERING SETS

In [17], the 5WH principle was proposed to summarize the information needs of a task and the sensing capabilities of a WSN, and in [28], the spatial relevancy of the provided information was introduced along with a way to measure it. In [29], a set of functions that model the sensing quality of each subset of nodes for each tasks are studied. Motivated by this prior work, we propose the relevancy of a sensor to a task as the degree to which the sensor can satisfy the task's QoI requirements. Specifically, we define:

$$r_{nm} = f\left(\underline{c}_n, \underline{q}_m\right) \in [0, 1], \quad \forall n \in \mathcal{N}, m \in \mathcal{M}, \tag{1}$$

where r_{nm} denotes the relevancy of sensor n to task m, and $f(\cdot)$ is a generic relevancy function that takes value in $[0, 1]$ by definition. A specific example of f is given in Section VII-A.

We define a sensor *irrelevant* to a task if and only if its relevancy to the task is 0. Examples of irrelevant sensors to a task include sensors whose sensing region have no overlap with the desired service region of a task, and sensors that cannot provide the type of information the task requires, such as a sensor providing temperature readings to an air pressure related task. On the other hand, for the coverage requirement of a task, we say a sensor *covers* a task if and only if the computed relevancy is 1. By definition, a sensor covers a task if and only if it can individually satisfy the desired QoI of the task. In an IoT sensory environment, the retrieved information from a single relevant sensor may not satisfy all QoI requirements of a task, thus resulting in a relevancy value that lies between 0 an 1. Therefore, to fully satisfy a task's QoI requirement, fusing information collected from multiple coordinating sensors will be needed.

A. INFORMATION FUSION

Some QoI requirements, like the coverage of a region, can be achieved by using a fusion algorithm (function) even if no individual sensor can meet the requirement. The authors in [28] propose to select a number of providers that cumulatively provide the most relevant information using an abstract, scalar-valued representation of QoI. While similar in principle, here we consider a more general way to accommodate a vector-valued QoI in information fusion. For ease of presentation, we use $g_{\mathcal{S}}(\cdot)$ for the generic fusion function, and clearly, g should take a variant number of single-sensor "capabilities" and output an aggregated capability in all aspects. Note that there are a number of works on information fusion in WSNs (see [30], [31]) that can be applied as a realization of $g_{\mathcal{S}}(\cdot)$; further elaboration of $g_{\mathcal{S}}(\cdot)$ is beyond the scope of this paper. Denoting the capability of a subset $\mathcal{S}$ of sensors by $\underline{c}_{\mathcal{S}}$, we have:

$$\underline{c}_{\mathcal{S}} = g_{\mathcal{S}}\Big(\{\underline{c}_n | n \in \mathcal{S}\}\Big). \tag{2}$$

Then, the relevancy of a subset of sensors to a task can be defined in the same way as that of a single sensor to a task based upon their aggregated sensing capability, i.e.,

$$r_{\mathcal{S},m} = f\left(\underline{c}_{\mathcal{S}}, \underline{q}_m\right), \quad \forall \mathcal{S} \subseteq \mathcal{N}, m \in \mathcal{M}. \tag{3}$$

B. CRITICAL COVERING SET

We define the *critical covering set* (CCS) of a task as a set of sensors whose aggregated sensor-to-task relevancy always achieves 1; and if the retrieved information from any sensor is lost, the aggregated relevancy will drop below 1. The calculation of CCS can be regarded as a set cover problem and can be solved by some greedy algorithms [32]. It is worth noting that there is finite probability that the sensors may not be able to cover the entire area of interest when randomly deployed. Furthermore, the desired QoI of certain tasks may be too demanding that even multiple collaborated sensors could not satisfy it. Therefore, it is possible that a task has no CCS. In this paper, we focus on the scenario in which there is sufficient density of deployed sensors to always guarantee the existence of CCSs for each task, with the realization that the system performance metric we defined in Section V-B also fits the scenario in which there exists no CCS for certain tasks. For ease for presentation, let $\mathbb{S}_m, \forall m \in \mathcal{M}$, be the set of all CCSs for task m and $\underline{\mathbf{S}} = \{\mathbb{S}_m\}$ the collection of all these sets.

V. QoI-AWARE ENERGY MANAGEMENT

As discussed earlier, in order to fully exploit the energy-efficiency in an IoT sensory environment without sacrificing the QoI delivered to a task, both (a) the irrelevant sensors, i.e., sensors that are not relevant to any future incoming tasks, and (b) the redundant sensors, i.e., sensors that are not critical to any tasks, are allowed to be switched to the sleeping mode (`OFF`). In this section, we propose a framework to control the duty-cycling of these sensors based upon the task model outlined in Section III.

A. DUTY-CYCLING OF SENSORS

The duty-cycle of a sensor is defined as the fraction of time that the sensor is `ON`, i.e., $\Sigma_n(T)/T, \forall n \in \mathcal{N}$, where $\Sigma_n(T)$ is the aggregation of the `ON` times during the lifetime T. We express the aggregated `ON` time as a function of T to explicitly describe its dependency on the lifetime. However, this straightforward definition of duty-cycle does not directly reflect the energy spent while switching between the two modes. Therefore, we propose a *generalized duty cycle* to explicitly incorporate the extra energy penalty paid each time the sensors switch modes. Specifically, let P_n^{sw}, P_n^{on} denote the amount of energy consumed when each time sensor switches modes and keeps awake, respectively, and $N_n(T)$ is the number of mode switchings sensor n makes. Then the generalized duty cycle η_n of sensor n is defined as

$$\eta_n(T) = \frac{P_n^{\mathrm{sw}}}{P_n^{\mathrm{on}}} \cdot \frac{N_n(T)}{T} + \frac{\Sigma_n(T)}{T}, \quad \forall n \in \mathcal{N}. \tag{4}$$

The goal of energy management is to minimize the generalized duty cycle in an IoT sensory environment, without sacrificing the QoI levels attained. At the beginning of each frame, the IoT EMS informs the gateway on the decisions as when to switch modes for sensors in the current frame. Let $\underline{A}(t) = \{a_n(t)\}$, $0 \le a_n(t) \le L$, $n \in \mathcal{N}$ denote the mode switching times of sensors in the frame following the decision at time t. When $a_n(t) < L$, the n-th sensor will switch mode at time $t + a_n(t)$ and, when $a_n(t) = L$, it will not switch mode in this frame. Clearly, the cardinality of the decision space of $\underline{A}(t)$ is L^N.

B. DELAY MODEL FOR TASKS

In practice, when the gateway sends the wake-up signals to the corresponding sensors they may not receive it immediately and be activated exactly at the scheduled time, mainly caused by the signal transmitting and processing latency. Moreover, even if the latency is very small to be neglected, it is likely that no active CCS of a task exists when task instances start. Therefore, the task may have to wait for the next frame when the EMS informs the gateway to wake up a CCS for its service, as shown in Fig. 3. Towards this end, we introduce the delay model for all tasks as follows.

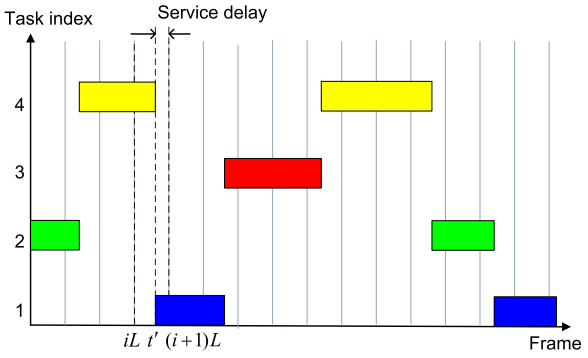

FIGURE 3. An illustrative example of service delay, where under the proposed task model, where four tasks follows different arrival and departure statistics, and during one single emergence, its maximum allowed delay tolerance is shown. Task 1, 2, and 4 have two instances and task 3 has only one instance.

We denote $d_{m,i}$ as the "attained" service delay of task m for its i-th instance. Note that this service delay may be tolerable, depending on the type of requested application (e.g., a few seconds delay for reporting the water quality levels are highly likely tolerable). We denote D_m as the maximum tolerable delay for any instance of task m, defined as the fraction of time compared to the lifetime of the task instance:

$$D_m = \tau_m l_{m,i}, \ \forall m \in \mathcal{M}, \ i \in \mathbb{N}^+, \tag{5}$$

where $l_{m,i}$ is the lifetime of task m's i-th instance, and $\tau_m \in [0, 1]$ is the *delay tolerance* of task m. Clearly, smaller τ_m represents more stringent delay requirement. Then, let $\zeta_{m,i}$ indicate if the system fails to satisfy task m's i-th instance's delay requirement; we have:

$$\zeta_{m,i} \triangleq \begin{cases} 0, & \text{if } 0 \le d_{m,i} \le \tau_m l_{m,i} \\ 1, & \text{if } d_{m,i} > \tau_m l_{m,i}, \end{cases} \tag{6}$$

$\forall m \in \mathcal{M}$, $i \in \mathbb{N}^+$. Define the number of instance of task m as I_m. Therefore, considering the overall service delay for a task, its average measured *delay failure* probability is

defined as:

$$\zeta_m = \frac{1}{I_m}\sum_{i=1}^{I_m}\zeta_{m,i}, \quad \forall m \in \mathcal{M},\ I_m \in \mathbb{N}^+, \tag{7}$$

and if the "attained" delay failure probability is smaller than a threshold ξ_m, we call its delay requirement is successfully satisfied:

$$\zeta_m \le \xi_m, \quad \forall m \in \mathcal{M}. \tag{8}$$

Hence, we have introduced the delay model for all tasks in a probabilistic manner with the task-specified parameters ξ_m (delay failure threshold) and τ_m (delay tolerance).

C. PROBLEM FORMULATION

At the beginning of each frame, the EMS informs the gateway of decisions made on the energy consumption state of each sensor $n \in \mathcal{N}$, i.e., which set of sensors should be waken up for task service in the current frame, and which set of sensors are allowed to be turned OFF, given the historical task evolution and sensor activity information, which we denote by $\underline{H}(t)$. For ease of presentation, we use Λ to denote a generic task evolution model without specifying the mathematical details. Therefore, a decision policy ν is defined as a mapping from $\underline{H}(t)$ to $\underline{A}(t)$, given the known task model and the pre-determined CCS information:

$$\underline{A}(t) = \nu(\underline{H}(t)|\Lambda, \underline{\mathbf{S}}). \tag{9}$$

The goal of EMS algorithm is to find the optimal decision policy ν^* that optimizes the sensor duty-cycles under the delay failure threshold for tasks. We propose a performance metric to describe the system performance, and then formulate the corresponding optimization problem.

1) MINIMIZE WEIGHTED AVERAGE DUTY CYCLE

As the task is changing from time to time, some sensors may be excessively depleted, which can greatly decrease the system lifetime. This model aims at minimizing the average duty cycle of the entire IoT sensory environment, while takes the energy consumption fairness into consideration. The optimization problem is:

$$\begin{aligned} &\underset{\nu}{minimize}:\ \overline{\eta} = \sum_{n\in\mathcal{N}}\beta_n\eta_n, \\ &subject to:\ \zeta_m \le \xi_m, \quad \forall m \in \mathcal{M}, \end{aligned} \tag{10}$$

where $\{\beta_n\} \in [0, 1]$ are weight factors, and $\sum_{n\in\mathcal{N}}\beta_n = 1$.

D. A GREEDY ALGORITHM

The optimization problem in (10) is generally NP-hard and their optimal solution are difficult to find without an exhaustive search. In this work, we propose a greedy algorithm for optimization problem (10). The algorithm is greedy in that at any decision point, it chooses the action that leads to the least *marginal increment* in $\overline{\eta}$.

We explicitly define the weight factors in (10) as the normalized ratio between the remaining energy $\bar{E}_n(t)$ and total energy reserve E_n to achieve a degree of energy fairness among all sensors:

$$\kappa_n(t) = \exp\left(-\frac{\bar{E}_n(t)}{E_n}\right), \quad \beta_n(t) = \frac{\kappa_n(t)}{\sum_{n=1}^{N}\kappa_n(t)}, \quad \forall n \in \mathcal{N}. \tag{11}$$

The mapping function is chosen to characterize the effect of decreased $\beta_n(t)$ with the increase of $\bar{E}_n(t)$. Clearly, $\beta_n(t)$ is a non-increasing function that the sensors with less residual energy are assigned with higher weights, indicating the smaller probability to be utilized at the decision moment. Therefore, a certain level of energy consumption fairness can be achieved, and network lifetime, defined as the time when the first sensor depletes its energy, is prolonged.

For ease of presentation, suppose the system starts at $t = 0$. Denote $t = iL$ as the beginning of the i-th frame, where $i \in \mathbb{N}$. Denote η_t^n as the *runtime* generalized duty cycle of sensor n up to time t. η_t^n can be updated recursively by

$$\begin{aligned} \eta_n(t) = \frac{1}{t}\Bigg[\frac{P_n^{\mathrm{sw}}}{P_n^{\mathrm{on}}}\Big(N_n(t) - N_n(t-L)\Big) + \Big(\Sigma_n(t) - \Sigma_n(t-L)\Big) \\ + \eta_n(t-L)\cdot(t-L)\Bigg], \end{aligned} \tag{12}$$

$t = iL, i \in \mathbb{N}^+$, with $\eta_n(0)$ defined to be zero. Note that $N_n(t) - N_n(t-L)$ and $\Sigma_n(t) - \Sigma_n(t-L)$ are the number of state switches and the aggregated ON time between time $t-L$ and time t for sensor n, respectively. We define the marginal increase in the normalized energy consumption of sensor n between time t and $t+L$ as:

$$\Delta_n(t) \triangleq \frac{P_n^{\mathrm{sw}}}{P_n^{\mathrm{on}}}\Big(N_n(t+L) - N_n(t)\Big) + \Big(\Sigma_n(t+L) - \Sigma_n(t)\Big), \tag{13}$$

and clearly, we have:

$$\eta_n(t) = \frac{1}{t}\Big(\Delta_n(t-L) + (t-L)\cdot\eta_n(t-L)\Big), \quad \forall n \in \mathcal{N}. \tag{14}$$

Further, define the weighted average marginal increase in the normalized energy consumption of all sensors between time t and $t+L$ as

$$\overline{\Delta}(t) \triangleq \sum_{n\in\mathcal{N}}\beta_n(t)\Delta_n(t). \tag{15}$$

At the i-th decision point, EMS needs to predict the task activities in the current frame and prepare the sensors accordingly. Rather than a global algorithm that minimizes $\overline{\eta}$ throughout the lifetime of the IoT sensory environment, we specify an algorithm that minimizes $\overline{\Delta}(t)$ at each decision point while satisfactorily guaranteeing the service delay requirement, i.e., the delay failure probability.

Specifically, define $\Phi_{m,i}(t)$ as the probability that the i-th instance of task m starts exactly at time t, and $\Psi_m(t)$ as the "preparation" probability that at least one CCS of task m exists at time t. Then, we can rewrite the measured delay failure probability in (7) as the sum of the probabilities when

CCS of a task is not prepared under the condition of the task appearance:

$$\zeta_m = \sum_{t=0}^{T} \Phi_{m,i}(t)\Big(1 - \Psi_m(t)\Big), \quad \forall m \in \mathcal{M}. \tag{16}$$

T denotes the task lifetime. Note that $\Phi_{m,i}(t)$ is zero almost everywhere. To see this, $\Phi_{m,i}(t) = 0$ if: (a) t is not a task transition time, (b) either less than $i-1$ or greater than $i+1$ instances of task m have occurred. In other words, $\Phi_{m,i}(t)$ takes non-zero value only at the time of task transition and task m is expecting its i-th instance. Therefore, the above summation is easy to compute.

The task transition is modeled as a (discrete) semi-Markov process. A semi-Markov process is a stochastic process which moves from one state to another, with the successive states visited forming a Markov chain, and that the process stays in a given state a random length of time (holding time). The state space of a semi-Markov process is countable and the distribution function of the holding times may depend on the current state as well as on the one to be visited next [33]. When modeling the task evolution by a semi-Markov model, the tasks are treated as the states. The behavior of the tasks can be summarized in the following three aspects:

- There is at most one task in service at any time slot. In the main context, we only consider the existence of task instance, and we discuss the inclusion of ''idle task'' in Section VIII;
- A new task starts immediately after a current task ends with certain ''task transition'' probability;
- The service time of a task is known at the time it starts.

We denote $\mathbf{P} = \{p_{k,m}\}$ as the task transition matrix, where $p_{k,m}$ is the transition probability from task k to task m. We also assume that $\mathbf{P}$ is known *a priori* to the gateway. In reality, $\mathbf{P}$ can be estimated based on the task evolution history by the EM algorithms[1] [34], which finds maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models in an iterative manner. The input of the EM algorithm is the observed data from task evolution history, consisting of the time of occurrence and departure of each task instance. More details will be given in Section VIII.

Lemma 5.1: For any given task m, its delay requirement is satisfied, if the probability of any instance of it fails to provide the satisfactory service delay τ_m, is upper-bounded by $\frac{\xi_m}{\pi_m}$, where ξ_m denotes the delay failure threshold, and π_m denotes the steady state probability of task m.

Proof: See Appendix A. ∎

In our model, the EMS knows exactly the time when the current task ends and the next task starts, whereas which specific task succeeds the current one is uncertain. If there is no transition between tasks in a frame, the system only needs to keep awake the CCS of the current task that leads to the least $\overline{\Delta}(t)$ and set the other sensors to sleep. Otherwise, if a task transition is bound to happen in a frame, the system has to wake up the corresponding sensors to make preparation for all possible succeeding tasks under their specific delay failure threshold. Suppose the current task will end during a frame and a new task will start at time t', where $t' \in [iL, (i+1)L)$. Our greedy algorithm at the i-th decision point is the solution to the following optimization problem

$$\begin{aligned} &\underset{\nu}{minimize}: \ \overline{\Delta}(iL) \\ &subject to: \ P_m(t')(1 - \Psi_m(t')) \le \xi_m, \quad \forall m \in \mathcal{M}. \end{aligned} \tag{17}$$

where $\overline{\Delta}(iL)$ is defined in (13) and (15), and $P_m(t')$ is the transition probability from the current task to task m.

To solve the above optimization problem, the constraint can be rewritten as

$$\Psi_m(t') \ge 1 - \min\left\{1, \frac{\xi_m}{P_m(t')}\right\}, \tag{18}$$

which illustrates a way of computing the preparation probability given the task transition model and delay failure requirement. Essentially, (18) specifies the minimum required probability of existence of CCSs for each task at the task transition time t'. Therefore, the EMS can determine whether to wake up a CCS for each task according to $\Psi_m(t')$, but the collective decisions for all tasks can be made either jointly or separately, since CCS may overlap for different tasks and a global optimum is achieved when the joint decision is made. However, this will induce further computational complexity, especially when M is large. Therefore, as another ''degree of greediness'', we let the EMS make decision separately for each task. After the decision on making preparation for which tasks is made, the EMS chooses and schedules the wake-up times for a subset of sensors that can cover that selected group of tasks and induces the minimum increase in the marginal energy consumption $\overline{\Delta}(iL)$. In the $(i+1)$-th frame, which task follows the previous task is already known and all the irrelevant sensors prepared for the possible occurrence of other tasks can be sent to sleep.

It is worth noting that signal propagation and processing latency has impact on the control decision operation, and we shall investigate this in Section VI. As for now, we assume that when the decision is made at EMS and further informs the gateway, the latter is able to control all corresponding sensors immediately, i.e., the wake-up time of the scheduled sensors is the task transition time t'. If the selected sensors can cover the next arrival task rightly after the completion of the current task, at the start moment of the next frame $(i+1)L$, non-critical sensors will be shut down. However, if the predicted sensors are incapable to cover the actual coming task, at time $(i+1)L$, the gateway sends a new wake-up signal and activates the sensors from the CCS which induces the minimum increase in the marginal energy consumption $\overline{\Delta}((i+1)L)$. This is because the gateway already knows the

[1]The expectation maximization (EM) algorithms will be executed within the energy management server (EMS).

identification of this task at time $(i+1)L$. It is easily obtained that, in the latter case, the service delay equals to $(i+1)L-t'$.

The algorithm can be summarized in the following steps:

1) At the beginning of each frame, shut down any sensor that is not critical to the current task, if there exist such sensors;
2) If no task transition is bound to happen in the current frame, keep the current status of each sensor until the next frame;
3) If a task transition is bound to happen in the current frame, for each task, compute the minimum required probability of existence of a CCS based on the delay failure threshold by (18), and determine (by random tests) whether to make preparation for that task according to the derived probability. At the time of task transition, wake up a subset of sensors that critically covers all the tasks to be prepared for, yet induces the minimum increase in the marginal energy consumption.

The algorithm is greedy in three aspects:

1) The algorithm satisfies the delay failure threshold every time task transitions happen;
2) The algorithm minimizes the marginal increase in energy consumption at every decision point;
3) The algorithm makes decision on whether to prepare for the possible occurrence of each task *separately*.

As discussed earlier, we can revise the greedy algorithm so that it makes decision on whether to prepare for the possible occurrence of each task *jointly*. Compared to the original one whose computational complexity increases linearly with M, the *revised greedy algorithm* increases exponentially with M and it is therefore much more time consuming. In order to show the potential improvement, we demonstrate the results for both greedy algorithms in Section VII.

VI. MODELING THE SIGNAL PROPAGATION AND PROCESSING LATENCY

In practice, because of the signal propagation and processing latency by MAC protocols and routing algorithms, the selected CCS members cannot be waken up immediately after the control decision is made. Furthermore, for many applications they do not require immediate task service, but allowing certain service delay after the specified start time. Towards this end, in this section, we improve our system model by explicitly considering the signal propagation and processing latency, modeled by certain amount of wake-up delays after the control decision is made when all CCS members have been successfully informed. In other words, this model consider the longest signal propagation delay from EMS to a CCS member, denoted as a period of ωL, $\omega \in \mathbb{Z}$. We also assume that once the wake-up signal has been sent, it cannot be revoked. In this section, we provide a thorough theoretical analysis on the new system model, coupled with experimental results of the impact of this signaling latency on average measured delay failure probability.

A. MODEL DESCRIPTION AND PROBLEM FORMULATION

Without loss of generality, we rename the appearance sequence of all task instances sequentially by index $k = 1, 2, \ldots$, as shown in Fig. 4. The figure also illustrates a localized view on a specific time period, where the k-th instance of all tasks arrives at time $t' \in (iL, (i+1)L)$, $i \in \mathbb{N}^+$, with its lifetime l_k. Further let x denote the interval between iL and t' as a random variable, then we have $t' = \sum_{i=1}^{k-1} l_i = iL + x$.

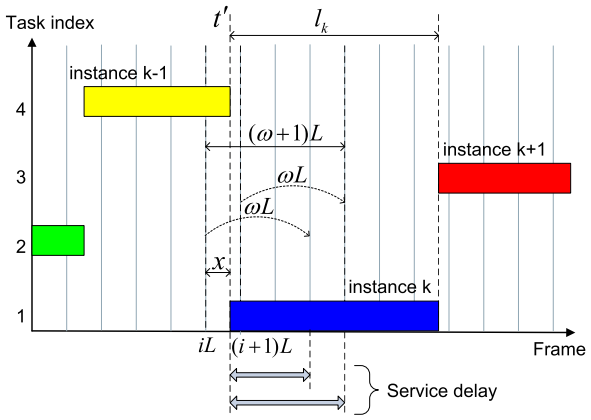

FIGURE 4. An illustrative example of service delay with signaling latency.

After receiving the control decision from EMS, the gateway sends a wake-up signal to the corresponding sensors at decision time iL. If at least one CCS of instance k exists at time t', the CCS members will be waken up at time $(i+\omega)L$, i.e., after this control decision successfully propagates to all sensors in ωL frames. Then, the service delay is computed as:

$$(i+\omega)L - t' = (i+\omega)L - (iL+x) = \omega L - x. \quad (19)$$

However, if no CCS of instance k exists at time t', by knowing exactly which task will appear at the next decision time $(i+1)L$, the gateway then sends a new wake-up signal which eventually takes effect at time $(i+1+\omega)L$, i.e., after considering the signal propagation and processing latency $(1+\omega)L$ from the current time. In this case, the service delay is:

$$(i+1+\omega)L - t' = (\omega+1)L - x. \quad (20)$$

Note that it is necessary to ensure $(\omega+1)L - x < l_k$, $\forall k \in \mathbb{N}^+$, since the task instance k needs to be handled before its termination. Consequently, $\omega < \lfloor l_{\min}/L \rfloor - 1$.

We are able to compute the probability that the service delay incurred at any instance is larger than the specified delay tolerance $\tau_m l_k$, conditioned when a CCS is well-prepared for service and otherwise, as:

$$P_m(t')\Psi_m(t')\Pr\{\omega L - x > \tau_m l_k\} + P_m(t')\left(1-\Psi_m(t')\right)\Pr\{(\omega+1)L - x > \tau_m l_k\} \le \xi_m, \quad (21)$$

$\forall m \in \mathcal{M}$. It is worth noting that (21) exactly characterizes the constraint of our optimization problem in (17), and more importantly a way to further theoretically derive $\Pr\{\zeta_{m,i} = 1\}$.

Theorem 6.1: Given the service time distribution in the previous assumptions, $\Psi_m(t')$ in (21) can be derived as:

$$\Psi_m(t') \geq \frac{F(\omega+1) - \frac{\xi_m}{P_m(t')}}{F(\omega+1) - F(\omega)}, \tag{22}$$

where

$$F(\omega) = 1 - \frac{\tau_m}{\mu L} \exp\left(-\frac{\omega\mu L}{\tau_m} + \mu l_{\min}\right)\left(\exp\left(\frac{\mu L}{\tau_m}\right) - 1\right). \tag{23}$$

Proof: See Appendix B. ■

We next discuss the feasibility issues of Theorem 6.1. As $F(\omega)$ monotonically increases with ω, we have $F(\omega+1) > F(\omega)$ always holds. Since $\Psi_m(t')$ denotes the *preparation probability* that at least one CCS of task m exists at time t', by definition it is a scaler between 0 and 1. Therefore, it requires:

$$F(\omega) \leq \frac{\xi_m}{P_m(t')} \leq F(\omega+1). \tag{24}$$

However, since $\xi_m \in [0, 1)$ is the maximum allowed delay failure probability, it is specified by applications, and does not have relations with the transition probability $P_m(t')$ from the current task to task m. Therefore, (24) may not always hold. Recall that $F(\omega)$ represents the probability that the incurred service delay is larger than the specified tolerance, or $F(\omega) \triangleq \Pr\{\omega L - x > \tau_m l_k\}$. Then, when $\frac{\xi_m}{P_m(t')} > F(\omega+1)$, we set $\Psi_m(t') = 0$. This is because if the maximum allowed delay failure is relatively very high or the task arrival probability is low enough, there is no need to prepare sensors for it. On the contrary, when $\frac{\xi_m}{P_m(t')} < F(\omega)$, we set $\Psi_m(t') = 1$ as a constant, indicating that if the maximum allowed delay failure is very low or a task instance is very likely to come at time t', it is necessary to prepare sensors for service.

In a summary, replaced by the new constraint in (22) under the realistic delay model, our objective function in (17) and proposed greedy algorithms can provide a sub-optimal solution. All other steps in Section V-D apply.

B. SATISFACTORY REGION OF DELAY TOLERANCE

The value of F denotes the theoretically derived probability that the service delay exceeds the maximum tolerable threshold. Fig. 5 shows the value of F with respect to (w.r.t.) different specified delay tolerance values, by varying ω and μL. Consistent with previous analysis, F monotonically increases with ω, since more severe signaling latency would result in higher delay outage probability. It also can be seen from the figure that higher delay tolerance τ_m leads to lower delay probability, and this probability increases with μL that characterizes the ratio of frame size and average duration of task instance. Higher μL (i.e., larger frame size or shorter instance duration) will relax the delay constraint imposed by tasks, and thus lower delay outage is expected.

As analyzed in Lemma 5.1, given task m, in order to successfully achieve its delay requirement, the probability of delay occurrence at any of its instances should be upper-bounded by ξ_m/π_m. Based on our analysis of signaling latency

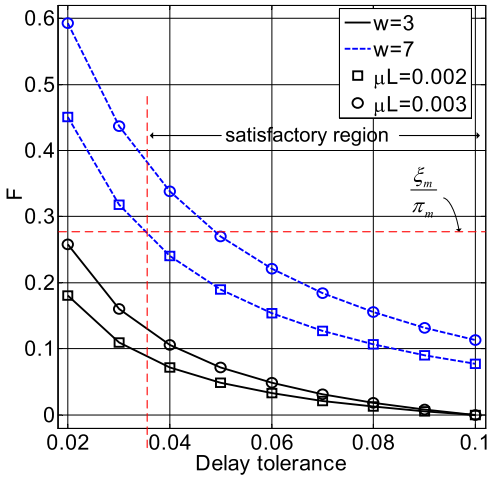

FIGURE 5. ***F* vs. delay tolerance threshold, varying parameters $\omega = 3, 7$ and $\mu L = 0.002, 0.003$.**

in (21), we rewrite the steady state form of Lemma 5.1 as:

$$\Psi_m F(\omega) + (1 - \Psi_m) F(\omega+1) \leq \frac{\xi_m}{\pi_m}, \tag{25}$$

where Ψ_m denotes the corresponding preparation probability under the steady state of task transitions. Since $F(\omega) < F(\omega+1)$, we relax the constraint in (25) by

$$F(\omega) < \frac{\xi_m}{\pi_m}, \quad \forall m \in \mathcal{M}. \tag{26}$$

According to the derivation of F as a non-increasing function of the delay tolerance τ_m, it is interesting to observe that τ_m cannot be arbitrarily chosen, but tightly coupled with system parameters L, ω, $\mathbf{P}_m$ and task parameters μ, $l_{\min}$. In Fig. 5, we visualize the condition (26) that eventually crosses all curves of different parameters. We call the region of τ_m satisfying the condition (26) as its *satisfactory region*. Therefore, given those parameters, the system has its own feasible working range, beyond which higher system settings (like L) should be configured. Deriving this lower-bounded region requires solving the transcendental equation and thus, numerical solutions are expected like the Newton's method.

VII. PERFORMANCE EVALUATION

In this section, we show an example of our methodology, by assuming a monitoring IoT application, such as using (randomly deployed) multi-functional sensors with certain sensing range to measure the water quality, temperature, air pollution or humidity of certain observation points. We present the system pertinent solutions, the environment settings and show the results.

A. SYSTEM MODEL AND SIMULATION SETUP

In our environmental monitoring system, each sensor with certain monitoring capability is randomly deployed and its spatial coverage follows a classic disk model. We assume that the sensory data within the sensing region is corrupted

by noise during measurement and/or transmissions. Fig. 6 shows an illustrative example of the sensing monitoring graph for sensor and task deployment, where $N = 15$ sensors are deployed in a 600×600 square area of unit distance, to obtain $M = 4$ different types of monitoring data for 4 specific locations (as red square), and a gateway is placed to collect the data from sensors.

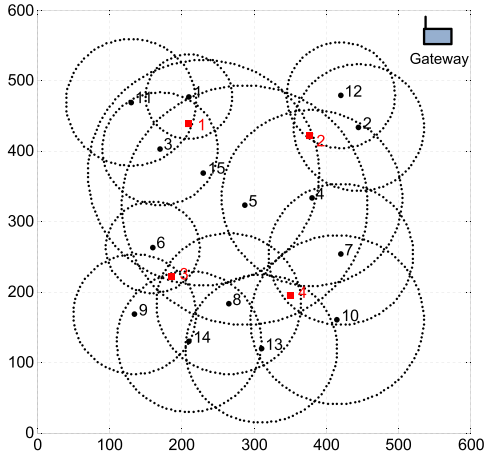

FIGURE 6. An illustrative example of the IoT application where 15 sensors with certain sensing range (as black dots) are randomly deployed to monitor the water quality, temperature, air pollution or humidity of four locations (as red square), and a gateway is placed to collect the data.

In this example, we consider measurement accuracy and service delay (both with multiple metrics) as two QoI requirements. For the former, we define its probabilistic model as:

$$\Pr\Big\{|Z_m(t) - z_m| \geq \delta_m\Big\} \leq \epsilon_m, \ \forall m \in \mathcal{M}, \tag{27}$$

where the random variable $Z_m(t)$ is the fused, sensor-retrieved information for task m at time t, and z_m is the actual but unknown information, i.e., the ground truth. Analogously to the desired QoI functions in [28], we define $\underline{q}_m$ as:

$$\underline{q}_m = \Big\{Y_m, (\delta_m, \epsilon_m)\Big\}, \ \forall m \in \mathcal{M}, \tag{28}$$

where Y_m and $\{\delta_m, \epsilon_m\}$ are the geographical location and accuracy requirement of task m, respectively. For service delay, tasks specify the required delay tolerance threshold τ_m and delay failure probability $\xi_m, \forall m \in \mathcal{M}$, as shown in Section V-B.

On the other hand, the sensing capability of sensor n, i.e., $\underline{c}_n$, can be defined as:

$$\underline{c}_n = \Big\{(X_n, r_n), \gamma_n\Big\}, \ \forall n \in \mathcal{N}, \tag{29}$$

where X_n is the location of the sensor and r_n is its sensing radius. We model the measurement noise as additive white Gaussian noise (AWGN) with variance γ_n for sensor n. A sensor-to-task relevancy function for this model is

$$\begin{aligned} f(\underline{c}_n, \underline{q}_m) &= f(X_n, r_n, \gamma_n, Y_m, \delta_m, \epsilon_m) \\ &= 1\{\text{dist}(X_n, Y_m) \leq r_n\} \\ &\quad \cdot \min\left\{\frac{\epsilon_m}{\Pr\{|W_n(t) - \omega L| \geq \delta_m\}}, 1\right\} \\ &= 1\{\text{dist}(X_n, Y_m) \leq r_n\} \cdot \min\left\{\frac{2\epsilon_m}{Q(\frac{\delta_m}{\sqrt{\gamma_n}})}, 1\right\}, \end{aligned} \tag{30}$$

$\forall n \in \mathcal{N}, m \in \mathcal{M}$, where $1\{statement\}$ is the indicator function that takes value 1 if the *statement* is true and 0 otherwise, $\text{dist}(X_n, Y_m)$ is the Euclidean distance between two points, the random variable $W_n(t)$ is the information retrieved from sensor n at time t, and $Q(\cdot)$ is the tail probability of the standard normal distribution.

If task m is serviced solely by sensor n, then $Z_m(t) = W_n(t)$; otherwise, if it is serviced by a subset $\mathcal{S}$ of sensors, then $Z_m(t) = W_{\mathcal{S}}(t)$, where $W_{\mathcal{S}}(t)$ is the fused information from a subset $\mathcal{S}$ of sensors. One possible information fusion algorithm of relevant sensors in this case can be:

$$W_{\mathcal{S}}(t) = \arg\min_w \frac{1}{|S|}\sum_{n\in\mathcal{S}}\frac{1}{\gamma_n}\Big|W_n(t) - w\Big|^2 = \frac{\sum_{n\in\mathcal{S}}\frac{W_n(t)}{\gamma_n}}{\sum_{n\in\mathcal{S}}\frac{1}{\gamma_n}}. \tag{31}$$

The right hand of (31) is a specific example of the fusion function $g_{\mathcal{S}}(\cdot)$ we defined in Section IV-A. Specially, if all γ_n are equal and $W_n(t) \sim \mathsf{N}(1, \gamma)$, the fused information of a group of K relevant sensors is the average of the individual ones and $W_t^{\mathcal{S}} \sim \mathsf{N}(1, \gamma/K)$, where $\mathsf{N}(\mu, \sigma^2)$ is a Gaussian distribution with mean μ and variance σ^2. Based on the above fusion algorithm, CCSs of every task can be computed during the sensor deployment stage, and used in the online duty-cycling control.

Our numerical result is based on the environmental monitoring system discussed above and is achieved in MATLAB. The capabilities (exclusive of sensing radius which is illustrated in Fig. 6) of all sensors are: $\gamma_n = 3, \forall n \in \mathcal{N}$. Moreover, $P_n^{\text{sw}} = 5$ and $P_n^{\text{on}} = 1, \forall n \in \mathcal{N}$ and the initial energy reserve of each sensor is set as 20,000 units. We assume that with the predetermined working power, each sensor is able to fully cover its sensing area. For all tasks in $\mathcal{M}$, the desired QoI satisfies: $\epsilon_m = 0.1$, $\delta_m = 1$, $\tau_m = 0$ (i.e., delay-sensitive applications). Assume that the service time of each task follows identical exponential distribution with average duration $1/\mu = 50$ time slots and minimum duration $l_{\min} = 25$ time slots (both are sufficiently longer than the frame size L), thus the arrival of all tasks is a Poisson process. A total of 1,000 task instances are simulated. The task transition matrix is given by:

$$\mathbf{P} = \begin{pmatrix} 0 & 1/10 & 2/5 & 1/2 \\ 1/5 & 0 & 3/5 & 1/5 \\ 1/3 & 1/3 & 0 & 1/3 \\ 4/5 & 1/10 & 1/10 & 0 \end{pmatrix}. \tag{32}$$

The sensor-to-task relevancy and CCSs can therefore be computed at offline, and there are 10 candidate sets in for

task 1, 2, and 4, and 20 CCSs for task 3. Meanwhile, each of CCS consists of three sensors.

We consider the optimization problem in (10) with the energy-aware weight factor β_n in (11), and the solution is obtained by using the proposed greedy algorithms outlined in Section V-D.

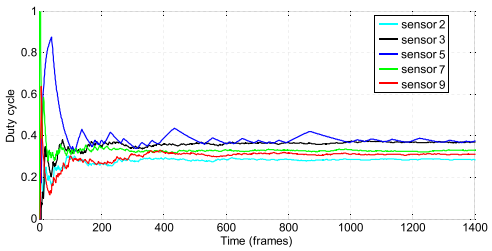

FIGURE 7. Sensor duty cycle *vs.* time.

B. SIMULATION RESULTS

In Fig. 7, we arbitrarily pick up five sensors and plot the evolving trend of their duty cycles over time. We set up the system parameters as stated above with $\xi_m = 0.04$, $\forall m \in \mathcal{M}$ and $L = 20$ time slots. It can be seen from the figure that after a few rounds of fluctuations at the very beginning when the sensors are trading-off their energy consumption with the provided QoI to tasks, the duty cycle of each sensor converges soon afterwards. This is because our proposed greedy algorithm successfully selects the best set of sensors for service under the stochastic, but Markovian task transitions, and the weight factors accurately capture the energy consumption state of all sensors and guarantees a degree of fairness among them.

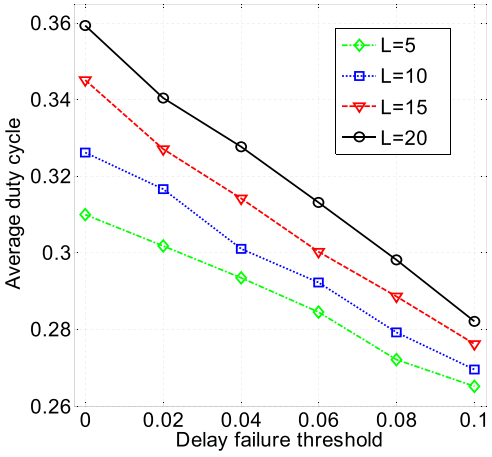

FIGURE 8. Average duty cycle *vs.* delay failure threshold, by changing frame size $L = \{5, 10, 15, 20\}$.

Next, we show the impact of two system parameters, the frame size L and sensor mode switching power P_n^{sw}, on the average duty cycle of all deployed sensors as shown in Fig. 8 and Fig. 9, respectively. The delay failure threshold ξ_m of all tasks are equally chosen while varying between 0 and 0.1.

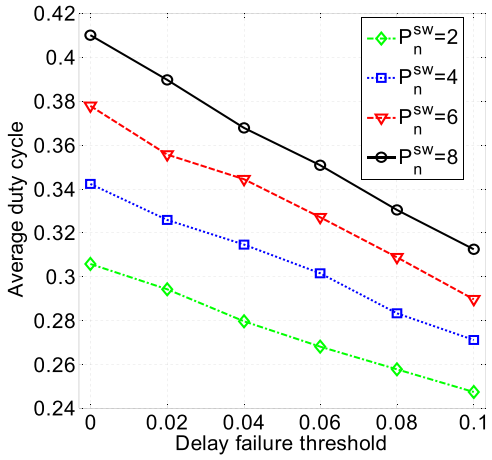

FIGURE 9. Average duty cycle *vs.* delay failure threshold, by changing switching power $P_n^{\text{sw}} = \{2, 4, 6, 8\}$.

We observe that for fixed L and P_n^{sw}, the average measured duty cycle linearly decreases with the increase of required delay failure threshold, as higher ξ_m relaxes the service delay requirement provided to all tasks by allowing certain amount of task instances to fail, and in turn the sensors allow to spend more time in the sleeping mode. For fixed ξ_m, the duty cycle increases with L and P_n^{sw}. Larger L represents the less frequency system control decisions and thus in order to provide satisfactory services to the next task, the system tends to wake up more than necessary sensors. These unnecessary sensors will stay awake until the next decision point when they can be turned `OFF` by the EMS. Clearly, the wasted `ON` times of sensors increase linearly with the frame length. Furthermore, larger P_n^{sw} indicates the less reluctant control behavior (or higher penalty) when switching the mode. Thus, the system decisions favor more those sensors who have been in the `ON` state, and let them continue servicing other tasks who may not eventually appear. Therefore, the energy consumption of all sensors are not optimally allocated, resulting in the larger both the average duty cycle and its variance.

Similar trends have been found when simulating the impact of task accuracy requirement ϵ_m, as shown in Fig. 10. With the increase of ϵ_m, i.e., less stringent QoI requirement that allows more measurement errors, the CCS of a task may involve less sensors for service, and in turn reducing the average duty cycle.

Then, we investigate the impact of system parameter L on the measured delay failure probability given different delay failure threshold, i.e., to judge if the required delay parameters are successfully guaranteed by the greedy algorithm. Table 2 shows the result, where the measured delay failure is satisfactorily less than the delay failure threshold and apparently, it increases with the threshold. However, it is interesting to observe that under the same delay failure threshold, the difference between measured results of different frame sizes L is insignificant. This can be explained by our setting of $\tau_m = 0$, $\forall m \in \mathcal{M}$, i.e., we only consider the delay-sensitive applications, and thus as long as the prepared sensors are

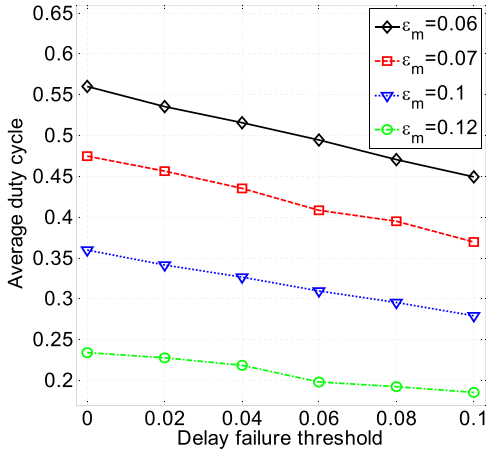

FIGURE 10. **Average duty cycle *vs.* delay failure threshold, by changing the task accuracy requirement $\epsilon_m = \{0.06, 0.07, 0.10, 0.12\}$.**

TABLE 2. **Average measured delay failure w.r.t. different frame sizes and delay failure thresholds.**

	$L = 5$	$L = 10$	$L = 15$	$L = 20$
$\xi_m = 0$	0	0	0	0
$\xi_m = 0.02$	0.0150	0.0133	0.0150	0.0137
$\xi_m = 0.04$	0.0338	0.0298	0.0295	0.0312
$\xi_m = 0.06$	0.0445	0.0428	0.0428	0.0468
$\xi_m = 0.08$	0.0655	0.0583	0.0637	0.0645
$\xi_m = 0.1$	0.0798	0.0755	0.0785	0.0795

incapable to service the coming task at the task transition time, the delay failure event is counted, irrespective how big the frame size L is (or when the next decision time will be, even if the sensors are well-prepared then). We shall investigate the impact of signal propagation and processing latency and frame size L on delay-tolerable applications in the next section.

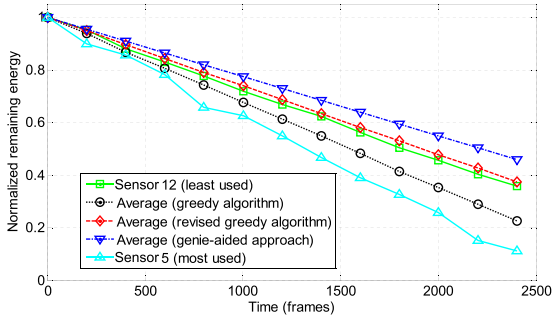

FIGURE 11. **Normalized remaining energy *vs.* time.**

Fig. 11 illustrates the energy depletion process, for $\xi_m = 0.04$, $L = 20$, with other parameters being set up as the former setting. Besides the proposed greedy algorithm and its revised version (i.e., jointly considering the CCSs of all tasks rather than treating them separately), we also show the result of the optimal solution where the EMS knows exactly which task succeeds the current one. The slope of a curve in the figure represents the energy depletion rate. We observe that the revised greedy algorithm only achieves a slightly better performance than the basic greedy algorithm, at the expense of more computational complexity. It can also be identified the gap between the greedy algorithm and the genie-aided optimal solution, due to the potential error in estimating the future arrived task. Furthermore, we plot the energy depletion process for two extremes: the least (sensor 12) and most used (sensor 5) sensor in the proposed greedy algorithm. As sensor 12 is located at the border area with limited sensing range, it can only cover task 2, which is also being covered by many other sensors like 2, 4, 5, 15, and thus being least frequently used. Meanwhile, sensor 5 is located at the center with a relatively larger sensing radius allowing it to service all four tasks, thus being used mostly. Nevertheless, the energy-aware weight factor in (11) that explicitly takes into the residual energy of a sensor into considerations, helps to lower the variance between these two extremes so that a certain degree of energy consumption fairness is achieved, as shown in the following.

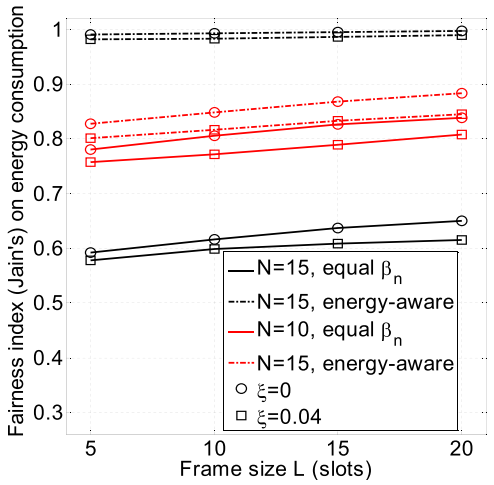

FIGURE 12. **Fairness index (Jain's) on energy consumption among all sensors.**

We investigate the energy consumption fairness, quantified by the Jain's fairness index[2] under the proposed greedy algorithm as shown in Fig. 12. We compare the proposed energy-aware weight assignment approach with the equal weight assignment, i.e., $\beta_n = 1/N$, $\forall n \in \mathcal{N}$. Clearly, for fixed number of sensors N, the Jain's index under energy-aware approach is higher than the one under equal assignment. Furthermore, when more nodes are deployed in a fixed geographic area, the increased node density helps to achieve better fairness among them since any single task would potentially be serviced by more CCSs. This trend does not hold for

[2]It is defined by $(\sum(E_n - \bar{E}_n))^2 / \left(N \sum(E_n - \bar{E}_n)^2\right)$, $\forall n \in \mathcal{N}$. The result ranges from $\frac{1}{N}$ (worst case) to 1 (best case). The larger the index is, the better fairness that we can achieve.

the equal weight setting, since the diversity gain cannot be utilized by assigning the same weights to all sensors irrespective their different amount of residual energy, and in turn the fairness level decreases with the number of sensors.

To investigate the impact of propagation delay, we still use the same task transition matrix $\mathbf{P}$ as shown in Section VII-A, and set $\xi_m = 0.1, \forall m \in \mathcal{M}$. A total of 1,000 task instances are simulated, with average duration of $1/\mu = 2000$ and minimum duration $\text{ł}_{\min} = 25$ time slots. Other parameters are the same as the basic setting in Section VII-A.

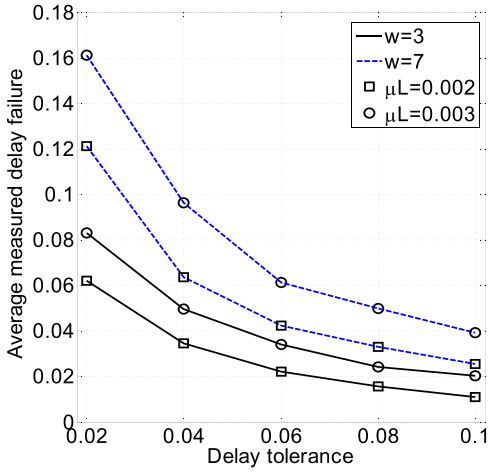

FIGURE 13. Average measured delay failure probability *vs.* delay tolerance threshold.

According to the proof of Lemma 5.1, the steady state $\boldsymbol{\pi}$ is obtained as (0.33, 0.14, 0.25, 0.28). Each element of the steady state denotes the stationary probability of a specific task. Fig. 13 shows the simulation result on the average measured delay failure among all tasks w.r.t. different delay tolerance thresholds. It can be observed that $\omega = 3$ can successfully guarantee that the required $\xi_m = 0.1$ for all $\tau_m \in [0.02, 0.1]$, however only part of entire τ_m values can satisfy the same requirement when $\omega = 7$; consistent with our analysis in Section VI-B. Furthermore, smaller parameter μL decreases the probability of delay failure occurrences either through more frequent control decisions (smaller L) or servicing longer instance duration (larger $1/\mu$); equivalently wider satisfactory region for delay tolerance given a ξ and $\mathbf{P}$.

VIII. IMPLEMENTATION GUIDELINES

We have made a few simplification and assumptions in Section III to ease the analysis, some of which may potentially generate new implementation guidelines in practice.

First, we assume that the CCS are known *a priori* to the EMS, which is usually realized during the deployment stage where the application owner deploys certain amount of devices to the specified network domain with known geographical locations. Then, the offline computation can be performed given the desired task location and requirements.

Second, the computation of task transition matrix requires advance algorithms like EM algorithm [34]. It uses an iterative procedure to compute the maximum likelihood estimation of a set of parameters in a given distribution (from empirical analysis). To apply it in our framework, EM needs the observed data from task evolution history, including the start and end time of each task instance, from which we can derive the transition times between tasks. Then, the EM algorithm approximates the parameters of the give distribution, as well as its expected value. Note that these expected values exactly represent the average number of transitions between each pair of two tasks, whose normalized values are a "noisy" version of the hidden, true value of all entries of the transition matrix $\mathbf{P}$.

Finally, in practical scenarios, tasks are described in human-friendly formats, e.g. XML/HTML, and thus a higher layer of format interpretation is used to translate the encoded scripts into the required QoI attributes, which can then be easily incorporated into our framework. Also, as a middleware bridging a variety of different applications and underlying networks, although tasks can be submitted randomly by each user, we assume that the aggregated behavior as the input to our platform exhibit some degree of determined characteristics, e.g., the total number of tasks, and their associated task QoI requirements. Moreover, there may not always be a ready task after the current task ends. Therefore, we can incorporate an "idle state" into the task transition model. The transition probability between idle state and other tasks can also be estimated by historical observations on task evolutions over time. When the idle task arrives, at the next decision point, we simply shutdown all prepared sensors and keep this state until the arrival of next task instance. Therefore, all previous analysis and proposed algorithms still apply.

IX. CONCLUSION

In this paper, a system-level efficient energy management framework is proposed to provide satisfactory QoI experience in IoT sensory environments. Contrary to past efforts, our proposal is transparent and compatible to lower protocols in use, and preserving energy-efficiency in the long run without sacrificing any attained QoI levels. Specifically, we introduced the new concept of QoI-aware "sensor-to-task relevancy" to explicitly consider the sensing capabilities offered by a sensor to the IoT sensory environments, and QoI requirements required by a task. Then, we proposed a novel concept of the "critical covering set" of any given task in selecting the sensors to service a task over time. Next, energy management decision is made dynamically at runtime, as the optimum for long-term traffic statistics under the constraint of the service delay. An extensive case study based on utilizing the sensor networks to perform environmental monitoring is given to demonstrate the ideas and algorithms proposed in this paper, and a simulation is made to show the performance of the proposed algorithms. To make our energy management framework more applicable and practical in realistic scenarios, we further considered the signal propagation and processing latency into our system model, and both theoretically and experimentally showed its impact on average

measured delay probability. Finally, based on our system model assumptions, we brought forward some implementation guidelines in practice and discussed the applicability of our proposal.

APPENDIX A
PROOF OF LEMMA 5.1

Define the steady state of transition matrix $\mathbf{P}$ of the Markov Chain that models the task evolution as $\boldsymbol{\pi}$. Due to the structure of the task model, all the possible states of the Markov Chain can be mutually visited. There exist a steady state $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_M)$ for all M tasks such that $\boldsymbol{\pi}\mathbf{P} = \boldsymbol{\pi}$, and $\sum_{m=1}^{M} \pi_m = 1$. This $\boldsymbol{\pi}$ can be found by solving the set of linear equations. Then, given any task m, its average measured delay failure probability of all instances in (7) can be rewritten as:

$$\begin{aligned}\zeta_m &= \lim_{I_m \to \infty} \frac{1}{I_m} \sum_{i=1}^{I_m} \zeta_{m,i} = \mathbb{E}\left[\zeta_{m,i}\right] \\ &= 1 \cdot \Pr\{\zeta_{m,i} = 1\}\pi_m + 0 \cdot \Pr\{\zeta_{m,i} = 0\}\pi_m \\ &= \Pr\{\zeta_{m,i} = 1\}\pi_m, \ \forall m \in \mathcal{M}. \end{aligned} \tag{33}$$

Therefore, for satisfactory delay performance under parameter ξ_m, we rewrite $\zeta_m \leq \xi_m$ as:

$$\Pr\{\zeta_{m,i} = 1\} \leq \frac{\xi_m}{\pi_m}, \quad \forall m \in \mathcal{M}, i \in \mathbb{N}^+. \tag{34}$$

APPENDIX B
PROOF OF THEOREM 6.1

As the service time of each task follows identical exponential distribution, the total number of instance occurrences has a Poisson distribution over $(0, t]$, and the occurrences are distributed uniformly on any interval of time. Therefore, the random variable x shown in Fig. 4 follows a uniform distribution in $(0, L)$. As l_k is exponentially distributed with average $1/\mu$ and lower-bound $l_{\min}$, its probability density function is given by:

$$f_l(l) = \begin{cases} \mu \exp(-\mu l + \mu l_{\min}), & l > l_{\min}, \\ 0, & \text{others}. \end{cases} \tag{35}$$

From (21), we have:

$$\Psi_m(t') \geq \frac{\Pr\{(\omega+1)L - x > \tau_m l_k\} - \frac{\xi_m}{P_m(t')}}{\Pr\{(\omega+1)L - x > \tau_m l_k\} - \Pr\{\omega L - x > \tau_m l_k\}}. \tag{36}$$

Let $F(w)$ denote the probability that the incurred service delay is larger than the specified tolerance, or $F(w) \triangleq \Pr\{\omega L - x > \tau_m l_k\}$, then:

$$\begin{aligned} F(w) &= \Pr\{l_k < \frac{\omega L - x}{\tau_m}\} \\ &= \int_0^L \frac{1}{L} \int_{l_{\min}}^{\frac{\omega L - x}{\tau_m}} \mu \exp(-\mu l_k + \mu l_{\min}) dl_k \, dx \\ &= 1 - \frac{\tau_m}{\mu L} \exp\left(-\frac{\omega \mu L}{\tau_m} + \mu l_{\min}\right)\left(\exp\left(\frac{\mu L}{\tau_m}\right) - 1\right). \end{aligned} \tag{37}$$

Hence, replace (37) back to (36), we obtain the closed-form expression for $\Psi_m(t')$ in (22).

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
[2] J. Zheng, D. Simplot-Ryl, C. Bisdikian, and H. T. Mouftah, "The internet of things," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 30–31, Nov. 2011.
[3] W. Wolf, "Cyber-physical systems," *Computer*, vol. 42, no. 3, pp. 88–89, Mar. 2009.
[4] G. Wu, S. Talwar, K. Johnsson, N. Himayat, and K. D. Johnson, "M2M: From mobile to embedded internet," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 36–43, Apr. 2011.
[5] Y. Zhang, R. Yu, W. Yao, S. Xie, Y. Xiao, and M. Guizani, "Home M2M networks: Architectures, standards, and QoS improvement," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 44–52, Apr. 2011.
[6] Y. Zhang, R. Yu, M. Nekovee, Y. Liu, S. Xie, and S. Gjessing, "Cognitive machine-to-machine communications: Visions and potentials for the smart grid," *IEEE Netw.*, vol. 26, no. 3, pp. 6–13, May/Jun. 2012.
[7] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.
[8] B. Hameed, I. Khan, F. Durr, and K. Rothermel, "An RFID based consistency management framework for production monitoring in a smart real-time factory," in *Proc. Internet Things*, Nov./Dec. 2010, pp. 1–8.
[9] MicroStrain, Inc. [Online]. Available: http://www.microstrain.com/, accessed Sep. 2013.
[10] X. Zhang, Y. Zhang, R. Yu, W. Wang, and M. Guizani, "Enhancing spectral-energy efficiency for LTE-advanced heterogeneous networks: A users social pattern perspective," *IEEE Wireless Commun.*, vol. 21, no. 2, pp. 10–17, Apr. 2014.
[11] C. Bisdikian, L. M. Kaplan, M. B. Srivastava, D. J. Thornley, D. Verma, and R. I. Young, "Building principles for a quality of information specification for sensor information," in *Proc. IEEE 12th FUSION*, Jul. 2009, pp. 1370–1377.
[12] R. Y. Wang and D. M. Strong, "Beyond accuracy: What data quality means to data consumers," *J. Manage. Inf. Syst.*, vol. 12, no. 4, pp. 5–33, 1996.
[13] M. E. Johnson and K. C. Chang, "Quality of information for data fusion in net centric publish and subscribe architectures," in *Proc. 8th Int. Conf. Inf. Fusion*, Jul. 2005, p. 8.
[14] Z. Sun, C. H. Liu, C. Bisdikian, J. W. Branch, and B. Yang, "QoI-aware energy management in Internet-of-Things sensory environments," in *Proc. 9th Annu. IEEE Commun. Soc. Conf. SECON*, Jun. 2012, pp. 19–27.
[15] N. Sadagopan, M. Singh, and B. Krishnamachari, "Decentralized utility-based sensor network design," *Mobile Netw. Appl.*, vol. 11, no. 3, pp. 341–350, 2006.
[16] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 61–72, Mar. 2002.
[17] C. Bisdikian, J. Branch, K. K. Leung, and R. I. Young, "A letter soup for the quality of information in sensor networks," in *Proc. IEEE Int. Conf. PERCOM*, Galveston, TX, USA, Mar. 2009, pp. 1–6.
[18] C. H. Liu, P. Hui, J. W. Branch, and B. Yang, "QoI-aware energy management for wireless sensor networks," in *Proc. IEEE Int. Conf. PERCOM Workshops*, Mar. 2011, pp. 8–13.
[19] C. H. Liu, C. Bisdikian, J. W. Branch, and K. K. Leung, "QoI-aware wireless sensor network management for dynamic multi-task operations," in *Proc. 7th Annu. IEEE Commun. Soc. Conf. SECON*, Boston, MA, USA, Jun. 2010, pp. 1–9.
[20] J. Ma, W. Lou, Y. Wu, X. Li, and G. Chen, "Energy efficient TDMA sleep scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 630–638.
[21] Y. Wu, X.-Y. Li, M. Li, and W. Lou, "Energy-efficient wake-up scheduling for data collection and aggregation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 2, pp. 275–287, Feb. 2010.
[22] R. Jurdak, P. Baldi, and C. V. Lopes, "Adaptive low power listening for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 8, pp. 988–1004, Aug. 2007.
[23] C. J. Merlin and W. B. Heinzelman, "Duty cycle control for low-power-listening MAC protocols," *IEEE Trans. Mobile Comput.*, vol. 9, no. 11, pp. 1508–1521, Nov. 2010.

[24] H. Yoo, M. Shim, and D. Kim, "Dynamic duty-cycle scheduling schemes for energy-harvesting wireless sensor networks," *IEEE Commun. Lett.*, vol. 16, no. 2, pp. 202–204, Feb. 2012.
[25] S. Zahedi, M. B. Srivastava, C. Bisdikian, and L. M. Kaplan, "Quality tradeoffs in object tracking with duty-cycled sensor networks," in *Proc. IEEE 31st RTSS*, Nov. 2010, pp. 160–169.
[26] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," *IEEE Commun. Mag.*, vol. 4, no. 44, pp. 115–121, Apr. 2006.
[27] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sensor Syst. (SenSys)*, Oct. 2003, pp. 171–180.
[28] G. Tychogiorgos and C. Bisdikian, "Selecting relevant sensor providers for meeting 'your' quality information needs," in *Proc. IEEE MDM*, Jun. 2011, pp. 200–205.
[29] A. Krause, A. P. Singh, and C. Guestrin, "Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies," *J. Mach. Learn. Res.*, vol. 9, pp. 235–284, Jun. 2008.
[30] E. F. Nakamura, F. G. Nakamura, C. M. S. Figueiredo, and A. A. F. Loureiro, "Using information fusion to assist data dissemination in wireless sensor networks," *Telecommun. Syst.*, vol. 30, nos. 1–3, pp. 237–254, 2005.
[31] E. Nakamura, A. A. F. Loureiro, and A. C. Frery, "Information fusion for wireless sensor networks: Methods, models, and classifications," *ACM Comput. Surv.*, vol. 39, no. 3, Sep. 2007, Art. ID 9.
[32] V. Chvatal, "A greedy heuristic for the set-covering problem," *Math. Oper. Res.*, vol. 4, no. 3, pp. 233–235, 1979.
[33] R. Pyke, "Markov renewal processes: Definitions and preliminary properties," *Ann. Math. Statist.*, vol. 32, no. 4, pp. 1231–1242, 1961.
[34] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc., Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

CHI HAROLD LIU (M'10) is currently a Full Professor with the School of Software, Beijing Institute of Technology, Beijing, China. He is also the Institute of Data Intelligence, the IBM Mainframe Excellence Center, Beijing, the IBM Big Data and Analysis Technology Center, and the National Laboratory of Data Intelligence for China Light Industry. He received the Ph.D. degree from Imperial College London, London, U.K., and the B.Eng. degree from Tsinghua University, Beijing, China. He was with IBM Research - China, Beijing, as a Staff Researcher and Project Manager, after working as a Post-Doctoral Researcher with Deutsche Telekom Laboratories, Berlin, Germany, and a Visiting Scholar with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. His current research interests include the Internet-of-Things (IoT), big data analytics, and wireless ad hoc, sensor, and mesh networks. He was a recipient of the Distinguished Young Scholar Award in 2013, the IBM First Plateau Invention Achievement Award in 2012, and the IBM First Patent Application Award in 2011, and was interviewed by EEWeb.com as the Featured Engineer in 2011. He has authored over 60 prestigious conference and journal papers, and holds over 10 European Union/U.S./Chinese patents. He serves as an Editor of the *KSII Transactions on Internet and Information Systems* and a Book Editor for five books published by Taylor & Francis Group, Boca Raton, FL, USA. He has also served as the General Chair of the IEEE SECON'13 Workshop on IoT Networking and Control, the IEEE WCNC'12 Workshop on IoT Enabling Technologies, and the ACM UbiComp'11 Workshop on Networking and Object Memories for IoT. He served as a Consultant of Asian Development Bank, Bain & Company, Boston, MA, USA, and KPMG, New York, NY, USA, and a Peer Reviewer of the Qatar National Research Foundation, and the National Science Foundation, China. He is a member of the Association for Computing Machinery.

JUN FAN is currently a Software Engineer with Baidu Inc., Beijing, China, and is currently pursuing the M.Sc. degree with the Department of Computer Science, Beijing Institute of Technology, Beijing. His research interests include cooperative communication, vehicular ad hoc networks, and the Internet-of-Things.

JOEL W. BRANCH (M'07) is currently a research staff member with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. His current research interests lie in application profiling and process automation for services supporting data center and cloud migration. He has also served as the Technical Assistant to the Director of Distributed Computing at IBM Research. He has authored numerous peer-reviewed papers on his work, and holds over 15 patents either issued or pending. In 2009, he was named as a Modern Day Technology Leader at the Black Engineer of the Year Awards. He received the B.S. degree in computer science from Howard University, Washington, DC, USA, in 2001, and the Ph.D. degree from the Rensselaer Polytechnic Institute, Troy, NY, USA, in 2007, where he also received an award for outstanding thesis work.

KIN K. LEUNG (F'01) received the B.S. degree from the Chinese University of Hong Kong, Hong Kong, in 1980, and the M.S. and Ph.D. degrees in computer science from the University of California at Los Angeles, Los Angeles, CA, USA, in 1982 and 1985, respectively. He started his career at AT&T Bell Labs, Murray Hill, NJ, USA, AT&T Labs, Austin, TX, USA, and Bell Labs of Lucent Technologies, New Providence, NJ, USA. Since 2004, he has been the Tanaka Chair Professor of Internet Technology with Imperial College London, London, U.K. His research interests include radio resource allocation, MAC protocol, TCP/IP protocol, mobility management, network architecture, real-time applications, and teletraffic issues for broadband wireless networks. He was a recipient of the Distinguished Member of Technical Staff Award from AT&T Bell Labs in 1994, and a co-recipient of the 1997 Lanchester Prize Honorable Mention Award. He was also a recipient of the Royal Society Wolfson Research Merit Award from 2004 to 2009.