# Energy-Efficient Scheduling of Internet of Things Devices for Environment Monitoring Applications

Taewoon Kim, Daji Qiao
Dept. of Electrical and Computer Engineering
Iowa State University, Ames, IA
{tkim, daji}@iastate.edu

Wooyeol Choi
Dept. of Computer Engineering
Chosun University, Gwangju, Republic of Korea
wyc@chosun.ac.kr

***Abstract*—The Internet of Things (IoT) paradigm has been proposed to assist and automate various activities such as environment monitoring by connecting physical devices in the area of our interest. Low-cost, battery-operated, and resource-limited IoT devices usually are densely deployed for robustness against node failures as well as for providing desired quality of monitoring. One of the effective ways to prolong the lifetime of an IoT network is to schedule selected IoT devices to enter the sleep mode and activate them later in a future time. However, this must be done carefully in order not to violate application-specific requirements. In this paper, we study the scheduling problem of IoT devices to prolong the network lifetime while satisfying both report-accuracy and timely-update requirements. We model and analyze the network as a Markov process, and derive system parameters. We formulate an optimal node activation scheduling problem and propose a low-complex greedy algorithm to expedite the scheduling process. Evaluation results demonstrate the effectiveness of both optimal and greedy algorithms.**

***Index Terms*—Internet of Things, sleep mode, node scheduling, Markov process**

## I. Introduction

Low-power, low-cost devices with sensing modules have been widely used in our daily lives as well as in special circumstances for particular missions such as environment monitoring and target tracking. Despite such devices' limited computing power and energy supply, they have become the essential component for Internet of Things (IoT) and Wireless Sensor Networks (WSNs). Such trend is expected to prosper by IEEE 802.11ah TG (Task Group) [1] that is intended to provide a unified wireless communication solution for large-scale IoT networks. To be specific, an 802.11ah-compatible AP (Access Point) is capable of associating with up to 8,192 devices with a coverage radius of about 1 km [2].

Although 802.11ah has built-in power saving methods [2], energy efficiency is still one of the biggest challenges for many IoT applications. This is because IoT devices (referred to as *nodes*) are battery-operated, and it may not be cost-effective or even possible to recharge or replace batteries in some applications, e.g., environment monitoring of an inaccessible terrain. In general, nodes are densely deployed in an IoT network for connectivity and coverage as well as for dealing with nodes' high failure rates [3]. Highly-populated nodes, however, increase the channel contention for the shared medium as well as produce redundant sensed readings, which decreases the overall energy efficiency and the network lifetime.

One effective way to prolong the network lifetime is to operate some of the nodes in the *sleep mode* and activate them later in a future time. However, the reduced number of active nodes may degrade the system performance, such as accuracy of the fusion report in environment monitoring applications. Also, the reduced number of active nodes may result in a violation of some application-specific requirements, such as maximum update interval or maximum delay requirement in environment monitoring applications.

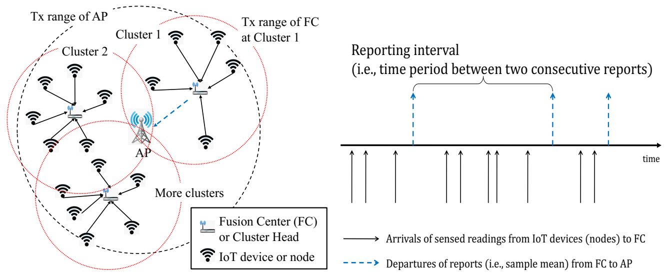
In this paper, we study the problem of energy-efficient activation/sleep scheduling of IoT devices to maximize the network lifetime for environment monitoring applications. Here, the network lifetime refers to the time period during which the network operates properly without violating any application-specific requirements. As shown in Fig. 1a, we consider a clustered IoT network with multiple fusion centers (FCs) serving as cluster heads to collect sensed readings from IoT nodes and then transmit the fusion reports to the AP. We consider the following two requirements: (1) [report-accuracy requirement] maximum error in the fusion report created by the FC, assuming that each sensed reading has an inherent measurement error; and (2) [timely-update requirement] maximum reporting interval from the FC to the AP.

To do so, we first model and analyze the network as a Markov process. Then, we compute the optimal number of nodes to activate, and configure the FC to satisfy the requirements. Finally, given the optimal number of nodes to activate, we formulate an optimal node scheduling problem to determine the set of nodes to activate by taking into consideration the nodal residual energy. The problem is formulated as a mixed integer linear program (MILP) problem that is intractable in general; thus, we also propose a low-complexity greedy algorithm to solve the problem.

The rest of this paper is organized as follows. The network model and problem formulation are given in Section II. We analyze the network in Section III. In Section IV, we derive an optimal node scheduling algorithm, and propose a practical greedy algorithm with a lower complexity. Evaluation results

The work reported in this paper was funded in part by U.S. National Science Foundation Grant 1730275, NRF grant funded by the Korean Government (MSIT) under Grant 2016R1C1B2009034, and by the National Program for Excellence in SW funded by the Korean Government (MIST) and IITP under Grant 2017-0-00137.

978-1-5386-3180-5/18/$31.00 ©2018 IEEE




(a) Network diagram with an AP, multiple clusters, and randomly deployed nodes.

(b) Illustration of arrivals of sensed readings to an FC (solid lines) and departures of reports from the FC (dashed lines).

Fig. 1. System model.

are presented in Section V. We summarize the related work in Section VI, and finally, conclude the paper in Section VII.

## II. System Model and Problem Overview

### A. Models and Assumptions

We study a clustered 802.11ah-compatible IoT network for environment monitoring applications. As shown in Fig. 1a, many battery-operated and resource-limited IoT devices (referred to as *nodes*) are deployed uniformly at random in the area to be monitored, while multiple FCs are deployed to relay the data from IoT devices to the AP in the form of fusion reports. Each FC has a stable power supply and can communicate directly with the AP. An FC associates with the nodes within its transmission range, forming a cluster. We use $\mathcal{W}$ to denote the set of nodes in a cluster. If a node is within the transmission range of multiple FCs, it associates with the one with the strongest signal strength.

An active node senses the environment (e.g., temperature, sound, light, or vibration) and transmits the sensed readings (referred to as *data*) to the FC. After collecting multiple sensed readings from its associated nodes, the FC takes a sample mean, and transmits only the mean value (referred to as *report*) to the AP. We use *reporting interval* to refer to the time interval between two consecutive reports from the FC to the AP, as illustrated in Fig. 1b.

*1) From IoT Devices to the Fusion Center:* We make the following assumptions on the sensed readings generated by IoT devices of the same cluster within the $l^{\text{th}}$ reporting interval: (1) they are generated according to a Poisson distribution with a rate of $\lambda$ which is the aggregate data rate of all nodes in the cluster ($\mathcal{W}$); (2) their values are i.i.d. (independent and identically distributed) random variables, with a mean of $\overline{D^l}$ and a variance of $\sigma^2$, where $\sigma^2$ can be viewed as the sensing/measurement error. We assume that nodes are homogeneous with inexpensive hardware and simple software; thus, they have the same measurement error of $\sigma^2$ and the same fixed data generation rate of $\lambda/|\mathcal{W}|$ on average. We assume each node has a limited amount of energy supply and without energy harvesting or energy replenishment capabilities. We assume that each FC has a buffer of finite size $B$, which can store up to $B$ number of data. All the received data are stored in the FC's buffer, till the FC takes a sample mean and transmits the fusion report to the AP. If the buffer becomes full, the FC discards any new data arrivals. Once the FC transmits a report to the AP, it flushes its buffer.

*2) From the Fusion Center to the AP:* We assume that the reports from the FC to the AP also follow a Poisson distribution with a rate of $\mu$. Different from IoT devices, the FC can adjust its reporting rate $\mu$ dynamically. We assume that $\mu$ is discrete, and it is adjustable between $[\mu_{\min}, \mu_{\max}]$ with a step size of $\Delta_\mu$.

Moreover, we assume that clusters are independent of each other. Thus, we focus our study on a single cluster in the rest of this paper. In 802.11ah, each node is assigned a unique identifier, called AID (association ID). Nodes are partitioned into different clusters based on their AIDs. The restricted access window (RAW) method – which is one of the most noteworthy features in 802.11ah – provides each cluster with a dedicated time interval to access the channel exclusively. By limiting the number of nodes that are allowed to compete for the shared wireless medium at the same time, the RAW scheme reduces channel contention, while increasing the period of time during which nodes can enter the sleep mode.

### B. Problem Statement

For each cluster, our **goal** is to schedule the activation of IoT nodes to maximize the lifetime of the cluster. Recall that the lifetime of a cluster refers to the time period during which the cluster operates properly without violating any requirements. Clearly, having too many active nodes may generate redundant sensed readings and deplete their battery fast. On the other hand, having too few active nodes may render the FC's report (i.e., sample mean) less accurate. In addition, even if the optimal number of nodes to activate is known, the schedule of node activations may greatly affect the lifetime of a cluster. In this regard, we formulate an energy-efficient node scheduling problem as shown below.

**Given**:

- the set of nodes $\mathcal{W} = \{w_1, w_2, \cdots, w_{|\mathcal{W}|}\}$ in a cluster,
- the total arrival rate $\lambda$ of sensed readings,
- the nodal residual energy $\phi(w_a)$, $\forall w_a \in \mathcal{W}$,
- the measurement error $\sigma^2$, and
- the buffer size $B$ at the FC,

with the following **constraints** (or application requirements):

- *report-accuracy requirement:* maximum expected error $\sigma_{req}$ in each fusion report,
- *timely-update requirement:* maximum expected reporting interval $\tau_{req}$,

the **outputs** of the proposed scheme are:

- the optimal reporting rate $\mu^*$ of the FC,
- the optimal number of nodes $\eta^*$ to activate, and
- the optimal node activation/sleep schedule,

such that the lifetime of a cluster is maximized. The **terminating condition** of the proposed method is: $|\{w_a : \forall w_a \in$

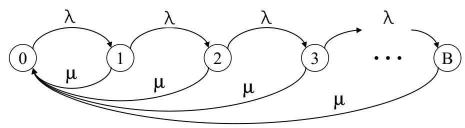



Fig. 2. Markov chain model for the FC (Fusion Center).

$\mathcal{W}$ such that $\phi(w_a) \geq \phi_{thr}\}| < \eta^*$, where $\phi_{thr}$ is the minimum residual energy for an IoT node to function normally. That is, when the number of nodes with enough residual energy is smaller than $\eta^*$, the proposed scheme terminates since the cluster can no longer satisfy the requirements.

### C. Overview of the Proposed Solution

To achieve this goal, we first model and analyze the network as a Markov process, based on which we compute the minimum number of nodes to activate ($\eta^*$) to satisfy the application requirements. Also, we compute the optimal $\mu^*$ at which the FC shall report to the AP. Then, given $\eta^*$, we derive an optimal scheme to schedule node activation/sleeping, by taking into consideration the residual energy of each node, so that we can iteratively determine the set of nodes to activate to maximize the cluster lifetime.

In this paper, the way to satisfy the requirements are based on the mathematical analysis, which helps to build a practical, robust solution. By considering the direct relation between the nodal residual energy and the cluster lifetime, the proposed solution performs better than methods that are based on the indirect relation instead (e.g., maximizing a utility function). Moreover, in the proposed solution, the FC can adapt its behavior (i.e., report rate $\mu$) so that it can adjust to the network dynamics to meet the goal and the requirements.

## III. Network Modeling and Analysis

Let $\{X_t\}_{t\geq 0}$ be a continuous-time stochastic process that takes a value from a countable set $\mathcal{B} = \{0, 1, 2, \cdots, b, \cdots, B\}$ at time $t \geq 0$. Here, $\mathcal{B}$ is a finite set since the FC discards any new arrivals when its buffer is full. By $X_t = b$ (i.e., $b$ is the state of $X_t$), we mean that the FC has $b$ number of data in its buffer at time $t$. When $X_t < B$, the state of the process is increased by 1, if there is a new arrival from an active node. The state of the process returns to 0 when the FC transmits a report to the AP. The amount of time spent in a state (referred to as *holding time*) is continuous, and the next state transition, given the present state, is independent of the past. Thus, $\{X_t\}_{t\geq 0}$ is a continuous-time Markov process with the holding time in each state being exponentially distributed. The corresponding continuous-time Markov chain (CTMC) is shown in Fig. 2.

Let $\mathcal{W} = \{w_1, w_2, \cdots\}$ be the set of nodes in a cluster. Arrivals from an active node to the FC is independent of arrvials from other nodes, and the total rate at which the CTMC transitions to the right is $\lambda$, given the current state is not $B$. On the other hand, the CTMC transitions to state 0 at rate $\mu$, given the current state is not state 0; please note that the FC does not report to the AP when it has no data in its buffer. Let $\mathbf{Q} = [q_{ij}]_{(i,j)\in\mathcal{B}\times\mathcal{B}}$ be the transition rate (or infinitesimal generator) matrix where $q_{ij}$ is the rate at which the CTMC transitions from state $i$ to state $j$.

$$\mathbf{Q} = \begin{bmatrix} -q_{01} & q_{01} & 0 & \dots & 0 \\ q_{10} & -(q_{10}+q_{12}) & q_{12} & \dots & 0 \\ q_{20} & 0 & -(q_{20}+q_{23}) & \dots & 0 \\ \vdots & 0 & 0 & \ddots & \vdots \\ q_{M0} & 0 & 0 & \dots & -q_{M0} \end{bmatrix}$$

Here, we have $q_{ii} = -q_i$ where $q_i = \sum_{\forall j\neq i} q_{ij}$ is the total outgoing rate from state $i$.

From the CTMC with the rate matrix, we can derive an embedded discrete-time Markov chain (DTMC) with exponential holding time. From DTMC, we know that all the states communicate with each other, forming one communication class, and thus, both DTMC and CTMC are irreducible. An irreducible DTMC with a finite state space is always recurrent (i.e., all states are recurrent), and so is the equivalent CTMC. Also, the CTMC is non-explosive (or regular) since the state space is finite. Since $\{X_t\}_{t\geq 0}$ is irreducible and recurrent, we can compute the stationary distribution $\boldsymbol{\pi} = \{\pi_1, \pi_2, \cdots, \pi_B\}$ by using both the global balance equation (i.e., $\sum_{\forall j\neq i} \pi_i \times q_{ij} = \sum_{\forall j\neq i} \pi_j \times q_{ji}$) and $\sum_{\forall i\in\mathcal{B}} \pi_i = 1$. Let $\rho = \frac{\lambda}{\lambda+\mu}$; then, we can get $\pi_i$ for all states $i$ as shown below, where $\pi_i$ is the proportion of time spent in state $i$ over a long period of time.

$$\pi_i = \begin{cases} \left\{\frac{1-\rho^B}{1-\rho} + \frac{\lambda}{\mu}\rho^{B-1}\right\}^{-1}, & \text{for } i = 0. \\ \rho^i\left\{\frac{1-\rho^B}{1-\rho} + \frac{\lambda}{\mu}\rho^{B-1}\right\}^{-1}, & \text{for } 0 < i < B. \\ \frac{\lambda}{\mu}\rho^{B-1}\left\{\frac{1-\rho^B}{1-\rho} + \frac{\lambda}{\mu}\rho^{B-1}\right\}^{-1}, & \text{for } i = B. \end{cases}$$

Given that $\{X_t\}_{t\geq 0}$ has a unique sum-up-to-1 stationary distribution, the process is positive recurrent.

By using the stationary distribution, we can compute the expectation of (1) the error of each sample mean, and (2) the reporting interval. Then, we can determine the minimum (i.e., optimal) number of nodes to activate to satisfy the corresponding requirements.

Let $U^l$ be a random variable for the number of buffered data that the FC uses to calculate the sample mean within a reporting interval $l$. For example, if $U^l = b \in \mathcal{B}$, it means that the FC has calculated and reported a sample mean with $b$ readings, and hence CTMC has made a transition from state $b$ to state 0. Then, we have $E[U^l] = \frac{1}{1-\pi_0}\sum_{i=1}^{B} i \times \pi_i$. The summation term is normalized by $1 - \pi_0$ since there is no transition from state 0 to itself. Let $D_m^l$ be a random variable for the value of the $m^{\text{th}}$ sensed reading that arrives at the FC within a reporting interval $l$. Since $D_m^l$'s are i.i.d. by assumption, the sample mean for the reporting interval $l$ is a random variable $S^l = \frac{1}{U^l}\sum_{m=1}^{U^l} D_m^l$. By the Central Limit Theorem, $S^l$ follows the Normal distribution with a mean of $\overline{D^l}$ and a variance of $\sigma^2/U^l$. Thus, the expectation of the sample mean error for the reporting interval $l$ is $\sigma/\sqrt{E[U^l]}$. The report-accuracy requirement requires that $\sigma/\sqrt{E[U^l]} \leq \sigma_{req}$.

The reporting interval is the time interval for the CTMC to start at state 0 and then return to state 0 for the first time. Please note that there is no direct transition from state 0 to itself. Let $\tau_{ii}$ be the smallest return time from state $i$ to state $i$, i.e., $\tau_{ii} = \inf\{t \geq 0 : X_t = i, X_{t-} \neq i | X_0 = i\}$. Since $\{X_t\}_{t\geq 0}$ is positive recurrent, we have $E[\tau_{ii}] = \frac{1}{\pi_i \times q_i} < \infty$. Thus, $\tau_{00}$ is the reporting interval. The timely-update requirement requires that $\tau_{00} \leq \tau_{req}$.

For a given $\mu$, as the number of active nodes increases, the expected error of each sample mean decreases since an FC can buffer more data during each reporting interval. Meanwhile, the expected reporting interval decreases because of a shorter holding time in state 0. Let $\eta^*_{ERR}(\mu)$ and $\eta^*_{RI}(\mu)$ be the minimum number of nodes to activate, in order to satisfy the report-accuracy and timely-update requirements, respectively, for a given $\mu$. Then, the minimum number of nodes to activate to satisfy both requirements can be computed as $\eta^*(\mu) = \max\{\eta^*_{ERR}(\mu), \eta^*_{RI}(\mu)\}$.

Furthermore, as $\mu$ increases, the expected reporting interval decreases. Thus, $\eta^*_{ERR}(\mu)$ has to increase to guarantee the FC buffers enough number of data within a shorter amount of time. On the other hand, as $\mu$ decreases, the expected reporting interval increases. Therefore, $\eta^*_{RI}(\mu)$ has to increase so that the FC can activate more nodes to reduce the holding time in state 0, in order to meet the maximum report interval requirement. To summarize, $\eta^*_{ERR}(\mu)$ is a non-decreasing function of $\mu$, while $\eta^*_{RI}(\mu)$ is non-increasing. Therefore, there exists at least one optimal $\mu$ that may yield the minimum number of nodes to activate. Let $\mu^*$ denote such an optimal $\mu$, and let $\eta^*$ be the smallest number of nodes to activate for $\mu^*$. When there are more than one $\mu$ producing the same optimal $\eta^*$, we choose the smallest one among the optima as $\mu^*$.

## IV. Node Scheduling Problem and Algorithms

Given $\eta^*$, we formulate an optimal node activation scheduling problem to maximize the network lifetime. In this paper, the maximum network lifetime is achieved by maximizing the lifetime of each cluster.

### A. Optimal Node Scheduling: Problem Formulation

Nodes are partitioned into a set of groups $\mathcal{G} = \{g_1, g_2, \cdots\}$, and the proposed scheme activates one group at a time. Here, we have $|\mathcal{G}| = G = \lfloor W/\eta^* \rfloor$, where $W = |\mathcal{W}|$. That is, the proposed scheme activates the minimum number ($\eta^*$) of nodes to maximize the energy saving. The lifetime of each group is determined by the node with the least amount of residual energy in the group. That is, the lifetime of $g_b$ is proportional to $\min\{\phi(w_a) : \forall w_a \in g_b\}$, where $\phi(w_a)$ is a function that returns the residual energy of node $w_a$. By maximizing the sum of minimum residual energy over all groups, we can form an optimal node scheduling problem as below (called P. 1).

$$\max_{\mathbf{Z}} \sum_{b=1}^{G} \min\{\phi(w_a) : \forall w_a \in g_b\} \tag{1a}$$

$$\text{subject to: } \forall b: \sum_{a=1}^{W} z_{ab} = \eta^* \tag{1b}$$

$$\forall a: \sum_{b=1}^{G} z_{ab} \leq 1 \tag{1c}$$

$$\forall a, b: z_{ab} \in \{0, 1\}, \tag{1d}$$

where $\mathbf{Z} = [z_{ab}]_{(a,b)\in\{1,2,\cdots,W\}\times\{1,2,\cdots,G\}}$ is a $W$-by-$G$ membership indicator matrix with $z_{ab} = 1$ if $w_a \in g_b$ or 0 otherwise (i.e., binary relation as in Eq. 1d). Each group $g_b$ consists of $\eta^*$ nodes (Eq. 1b), and each node $w_a$ belongs to at most one group (Eq. 1c). We, then, transform P. 1 into an MILP formulation as shown below (called P. 2).

$$\max_{\mathbf{Z},\overline{\mathbf{Z}},\mathbf{v}} \sum_{b=1}^{G} v_b \tag{2a}$$

$$\text{subject to: constraints in (1b), (1c), (1d)}$$

$$\forall a, b: v_b \leq \phi(w_a) \times z_{ab} + \phi_{max} \times \overline{z}_{ab} \tag{2b}$$

$$\forall a, b: z_{ab} + \overline{z}_{ab} = 1, \tag{2c}$$

where $\overline{\mathbf{Z}} = \mathbf{1}_{W\times G} - \mathbf{Z}$ (Eq. 2c) and $\phi_{max}$ is the maximum battery capacity of a node. From both Eq. 2a and Eq. 2b, we have $v_b = \min\{\min\{\phi(w_a) : \forall w_a \in g_b\}, \phi_{max}\}$. The new objective function (Eq. 2a) is equivalent to the previous one (Eq. 1a) as long as each group has at least one node, which is always guaranteed by Eq. 1b. The optimal solutions $\mathbf{Z}^*$ and $\mathbf{v}^* \in \mathbb{R}^G$ indicate which node belongs to which group and the set of the group-wise least residual energy, respectively. The number of binary variables in P. 2 is $W \times G \approx W^2/\eta^*$, which could make the problem intractable as the number of nodes increases. However, by carefully studying P. 2, we can reduce the number of binary variables.

Let $\Phi = \{\phi_{[1]}, \phi_{[2]}, \phi_{[3]}, \cdots, \phi_{[W]}\}$ be an ordered list of $\phi(w_a)$ for $a = 1, 2, \cdots, W$, such that $\phi_{[k]} < \phi_{[k+1]}$.[1] Also, let $g^{[k]}$ be a group that the minimum residual energy among the nodes in the group is the $k^{\text{th}}$ smallest among those values of all groups. For $g^{[k]}$, the set of residual energy of the nodes in the group is denoted by $\Phi^{[k]} = \{\phi^k_{[1]}, \phi^k_{[2]}, \cdots, \phi^k_{[\eta^*]}\}$ such that $\phi^k_{[l]} < \phi^k_{[l+1]}$. For the minimum residual energy, $\phi^k_{[1]}$, of groups $g^{[k]}$ for all $k = 2, 3, \cdots, G-1$, we have $\phi^{k-1}_{[1]} < \phi^k_{[1]} < \phi^{k+1}_{[1]}$.

**Claim 1 (Optimal Node Partitioning)**:[2] Given that Eq. 2a equals $\max \sum_{b=1}^{G} \phi^b_{[1]}$, we claim that an optimal partitioning of $\phi$ values[3] (or an optimal node partitioning) is: $\hat{\Phi}^{[1]} = \{\phi_{[k+1]}, \phi_{[k+2]}, \cdots, \phi_{[k+\eta^*]}\}, \hat{\Phi}^{[2]} = \{\phi_{[k+\eta^*+1]}, \phi_{[k+\eta^*+2]}, \cdots, \phi_{[k+2\eta^*]}\}, \cdots, \hat{\Phi}^{[G]} = \{\phi_{[k+(G-1)\eta^*+1]}, \phi_{[k+(G-1)\eta^*+2]}, \cdots, \phi_{[k+G\eta^*]}\}$, where $k = W - \eta^* \times \lfloor \frac{W}{\eta^*} \rfloor$. The corresponding optimal $\hat{\mathbf{v}}$ vector is: $[\hat{\phi}^1_{[1]}(= \phi_{[k+1]}), \hat{\phi}^2_{[1]}(= \phi_{[k+\eta^*+1]}), \cdots, \hat{\phi}^G_{[1]}(= \phi_{[k+(G-1)\eta^*+1]})]^T$. Also, $\hat{\mathbf{v}}$ is element-wise greater than or equal to any other $\mathbf{v} = \{\phi^1_{[1]}, \phi^2_{[1]}, \cdots, \phi^G_{[1]}\}$, and thus, we have $\sum_{b=1}^{G} \hat{\phi}^b_{[1]} \geq \sum_{b=1}^{G} \phi^b_{[1]}$.

Given the optimal partitioning of nodes, we can make P. 2 more tractable by scheduling one group at a time,

[1] We assume $\phi_{[k]}$ is real-valued. Thus, without loss of generality, we assume that $\phi_{[k]} \neq \phi_{[k']}$ if $k \neq k'$.

[2] The proof of **Claim 1** is omitted due to space limitation.

[3] Since each $\phi$ value is unique, we can derive the partitioning of nodes from the partitioning of $\phi$ values.

thus reducing the number of binary variables. The modified problem P. 3 chooses $\eta^*$ nodes to form a group $\hat{g}^{[G]}$, which is the set of nodes with residual energy being $\hat{\Phi}^{[G]}$.

$$\max_{\mathbf{z},\overline{\mathbf{z}},v} \; v \tag{3a}$$

$$\text{subject to:} \; \sum_{a=1}^{W} z_a = \eta^* \tag{3b}$$

$$\forall a: \; z_a \in \{0,1\} \tag{3c}$$

$$\forall a: \; v \le \phi(w_a) \times z_a + \phi_{max} \times \overline{z_a}, \tag{3d}$$

$$\forall a: \; z_a + \overline{z}_a = 1, \tag{3e}$$

where $\mathbf{z} \in \{0,1\}^W$ denotes the membership relation between each node and $\hat{g}^{[G]}$, $\overline{\mathbf{z}} = \mathbf{1}_W - \mathbf{z}$, and $v \in \mathbb{R}$ is the minimum nodal residual energy in $\hat{g}^{[G]}$, i.e., $\hat{\phi}^G_{[1]}$ or $\phi_{[k+(G-1)\eta^*+1]}$. In P. 3, the number of binary variables is $W$, which is much smaller than $W^2/\eta^*$ for a large $W$ and a small $\eta^*$.

To summarize, we have formulated an optimal node scheduling problem P. 1 which partitions all nodes in a cluster into $G$ groups. The problem is then converted to an MILP formulation which is P. 2. Finally, based on the analytical solution to P. 2 given by **Claim 1**, we have formulated an optimal node scheduling problem P. 3 which chooses only $\eta^*$ number of nodes to activate to reduce the complexity of the combinatorial optimization problem.

### *B. Optimal and Greedy Node Scheduling Algorithms*

Given the optimal node scheduling for $\hat{g}^{[G]}$ from P. 3, we can derive an Optimal Node Scheduling (ONS) algorithm which runs as follows. [Step 1] ONS runs P. 3 to choose $\eta^*$ number of nodes to activate. [Step 2] When any of the active nodes does not have enough residual energy to operate, remove/deactivate the node, and go back to [Step 1]. Please note that P. 3 is still an MILP. When there are a small number of binary variables (e.g., $< 50$), an optimal solution can be found in a negligible amount of time, but it is not always the case if the number of nodes in a cluster is large or the processing power of the FC is limited.

To further reduce the complexity, we propose a Greedy Node Scheduling (GNS) algorithm that produces the same node selection as ONS but with a much lower complexity of O($\eta^* W$). Given that an optimal node selection for $\hat{g}^{[G]}$ is equivalent to finding nodes with residual energy of $\hat{\Phi}^{[G]}$ = $\{\phi_{[k+(G-1)\eta^*+1]}, \phi_{[k+(G-1)\eta^*+2]}, \cdots, \phi_{[k+G\eta^*]}\}$, this can be done by choosing $\eta^*$ nodes with the highest residual energy. Please note that $\phi_{[k+(G-1)\eta^*+1]}, \phi_{[k+(G-1)\eta^*+2]}, \cdots,$ $\phi_{[k+G\eta^*]}$ correspond to $(\eta^*)^{\text{th}}$, $(\eta^*-1)^{\text{st}}$, $\cdots$, $1^{\text{st}}$ highest nodal residual energy, respectively, in a cluster. GNS has the same two-step algorithm as ONS except in [Step 1], instead of solving P. 3, GNS iteratively selects a node with the $n^{\text{th}}$ highest nodal residual energy where $n = 1, 2, \cdots, \eta^*$ to activate. The algorithm for ONS and GNS is given in Algo. 1.

Algo. 1 first checks if the number of nodes in the cluster is at least $\eta^*$ (line 1); if not, it terminates (line 2). Then, the set $g$ is initialized to an empty set (line 4). For ONS, $g$ is determined by the solution to P. 3 (lines 5–7). On the other hand, GNS chooses nodes to activate by iteratively searching for a node with the highest residual energy (lines 8–12). At line 10, the $\mathtt{idx}(\cdot)$ function returns the index set; e.g., $\mathtt{idx}(\{w_1, w_2, w_7\}) = \{1, 2, 7\}$. When $g$ is determined, all nodes in the group become activated (line 15), while the others stay in the sleep mode to minimize the energy consumption. The $\phi_{thr}$ (lines 16) is the battery threshold such that if the residual energy of a node drops below the threshold, the node can no longer function properly. Any battery-drained nodes will be removed from both $g$ and $\mathcal{W}$ (line 17), and the algorithm starts over (line 20) from the beginning.

**Algorithm 1** Optimal/Greedy Node Scheduling

1: **if** $|\mathcal{W}| < \eta^*$ **then**
2:    Terminate // infeasible, not enough nodes
3: **end if**
4: $g \leftarrow \{\}$ // initialize to an empty set
5: **if** ONS is chosen **then**
6:    $\mathbf{z}^* \leftarrow$ Solve P. 3
7:    $g \leftarrow g \cup \{w_a\}, \forall a$ such that $z^*_a = 1$
8: **else if** GNS is chosen **then**
9:    **while** $|g| < \eta^*$ **do**
10:       $a^* \leftarrow \arg_a \max\{\phi(w_a) : \forall a \in \mathtt{idx}(\mathcal{W} - g)\}$
11:       $g \leftarrow g \cup \{w_{a^*}\}$
12:    **end while**
13: **end if**
14: **while** $|g| = \eta^*$ **do**
15:    Activate nodes in $g$
16:    **if** $\exists w_a \in g$ such that $\phi(w_a) < \phi_{thr}$ **then**
17:       $g \leftarrow g - \{w_a\}$, $\mathcal{W} \leftarrow \mathcal{W} - \{w_a\}$
18:    **end if**
19: **end while**
20: Go to line:1

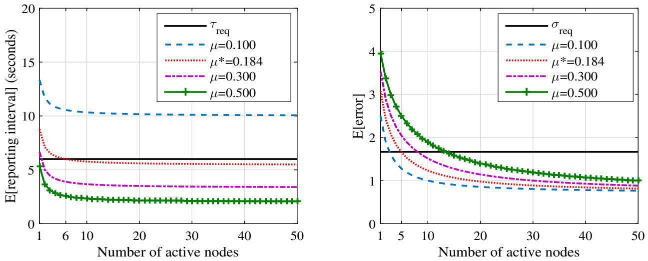
## V. Performance Evaluation

We have implemented five different node activation scheduling algorithms and compared their performances with each other: ONS (Optimal Node Scheduling), GNS (Greedy Node Scheduling), RNS (Random Node Scheduling), SNS (Sequential Node Scheduling), and invGNS (inverse GNS). RNS activates nodes uniformly at random, SNS activates nodes with the smallest AID[4] first (which is similar to *first-come, first-served*), and invGNS activates nodes with the least amount of nodal residual energy first, which is the exact opposite of GNS. All five algorithms activate $\eta^*$ number of nodes at a time. We used Matlab [4] to solve the optimization problem P. 3 as well as to implement a simulator and the five algorithms.

### *A. Simulation Setup*

Unless stated otherwise, we use the following default simulation configuration. The FC is associated with $|\mathcal{W}| = 50$ nodes, and can buffer up to $B = 50$ data. Each node has a fixed rate of $0.3 = \lambda/|\mathcal{W}|$ (data per second, from the node

[4]As a reminder, when a node joins a 802.11ah-compatible network, it is assigned a unique identifier, called AID.




(a) Reporting interval decreases as the number of active nodes increases for each given $\mu$.

(b) Error in a report decreases as the number of active nodes increases for each given $\mu$.

Fig. 3. Expectation of the reporting interval and report error with respect to the number of active nodes.

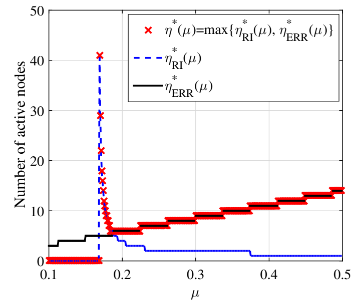



Fig. 4. Optimal number of nodes $\eta^*(\mu)$ to activate with respect to $\mu$, where $\eta^*_{RI}(\mu)$ and $\eta^*_{ERR}(\mu)$ are the smallest number of nodes to activate to satisfy reporting interval and report error requirements, respectively, and $\mu^* = 0.184$.

to the FC). The variance of the sensed readings (i.e., the measurement error) is $\sigma^2 = 25$. We assume the following requirements: $\sigma_{req} = \frac{1}{3}\sigma$, and $\tau_{req} = 6$ seconds (i.e., the AP expects at least one report every six seconds on average). For $\mu$ (reports per seconds, from the FC to the AP), we have $\mu_{\min} = 0.1$, $\mu_{\max} = 0.5$, and $\Delta_\mu = 0.001$. Considering that the radio module is one of the major sources of energy consumption [5], we assume the following simplified energy model. At the beginning, each node is assumed to have a random amount of energy drawn from a uniform distribution, $U(0, \phi_{\max}]$, where $\phi_{\max} = 100$. Each node consumes one unit of power for generating one sensed reading or transmitting one sensed reading to the FC; in this regard, $\phi_{thr}$ is set to one. All results are averaged over 50 simulation runs. We also have measured the 95% confidence intervals of the results, which, however, are omitted from the figures as they are very small.

### *B. Identification of Optimal $\mu^*$ and $\eta^*$*

We first show how to find the optimal $\mu^*$ and $\eta^*$. The analytical results in Fig. 3a and Fig. 3b show the expected reporting interval and report error, respectively, as the number of active nodes increases. As shown in Fig. 3a, with an increased number of active nodes, the expected reporting interval decreases for each given $\mu$. This is because the FC spends less and less time in state 0 as the total incoming data rate increases. The reporting interval $\tau_{00}$ is the sum of the holding time in state 0 (denoted by $\tau_0$) and the sum of holding times in non-zero states (denoted by $\tau_\Sigma$) until the CTMC returns to the state 0 for the first time. While $\tau_\Sigma$ is a function of both $\mu$ and $\lambda_{act}$, where $\lambda_{act} =$ (the number of active nodes) $\times \lambda/|\mathcal{W}|$, $\tau_0$ is dependent on $\lambda_{act}$ since there is no transition from state 0 to itself. As the number of active nodes increases, so does the rate of data arrivals from nodes to the FC. Thus, the waiting time until the first data arrival also decreases, resulting in the reduced reporting interval.

As shown in Fig. 3a, if $\mu$ is too small, e.g., 0.100, the reporting interval requirement cannot be satisfied. On the other hand, for larger $\mu$'s, we can identify the smallest number of active nodes, beyond which the reporting interval requirement can always be satisfied. Similarly, as shown in Fig. 3b, the report error decreases as the number of active nodes increases. This is because, during each reporting interval, the FC collects more and more data from nodes as $\lambda_{act}$ increases.

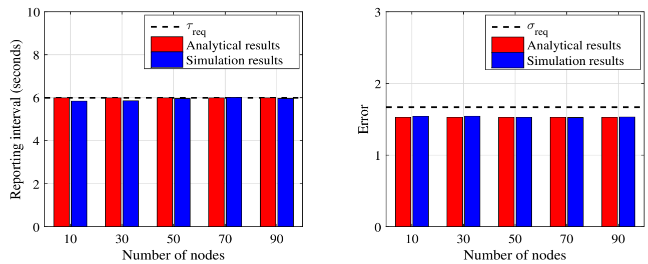
Since both reporting interval and report error requirements have to be satisfied at the same time for a given $\mu$, we first take the smallest number of active nodes from each requirement, and then, take the larger one to find $\eta^*(\mu)$ for a given $\mu$, i.e., $\eta^*(\mu) = \max\{\eta^*_{ERR}(\mu), \eta^*_{RI}(\mu)\}$. The trace of $\eta^*(\mu)$ for $\forall\mu \in [\mu_{\min}, \mu_{\max}]$ is shown in Fig. 4. For small values of $\mu \in [\mu_{\min}, 0.168]$, the FC cannot satisfy the reporting interval requirement, and thus, $\eta^*$ is set to 0 to signify infeasibility. When $\mu > 0.168$, there exists a feasible $\eta^*(\mu)$ for each $\mu$, and in the end, we have found that $\mu^* = 0.184$ to yield the smallest $\eta^* = 6$. Note that the reporting interval requirement is the dominating factor in determining $\eta^*$ for $\mu^* = 0.184$; this can be observed from Fig. 3.

### *C. Validation of the Proposed Markov Model*

To validate the Markov model we used, we compare the results from the analytical model with the simulation results, as shown in Fig. 5. It is clear that the simulation results, i.e., reporting interval in Fig. 5a and report error in Fig. 5b, are close to the analytical results, which validates our analytical model of the network. Since the reporting interval requirement is the dominating factor when determining $\eta^*$, expected reporting interval is very close to $\tau_{req}$, while there is a gap between expected report error and $\sigma_{req}$.

### *D. Performance Comparison*

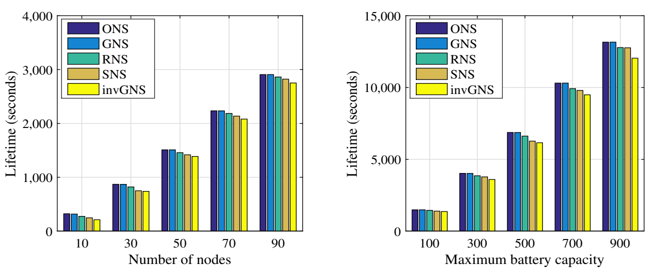
We also have compared the cluster lifetime with respect to the number of nodes in a cluster, and the results are plotted in Fig. 6a. As shown in the figure, both ONS and GNS result in the longest lifetime than other schemes. Also, the performance of GNS equals that of ONS with a much lower complexity. The invGNS has the worst performance as it activates node in the exactly opposite way of the GNS. The RNS outperforms SNS, showing that the randomization in node activation yields a better performance than both SNS and invGNS. In addition, we have compared the cluster lifetime with different maximum battery capacity when $|\mathcal{W}| = 50$. That is, the initial nodal




(a) Reporting intervals.

(b) Errors in a report.

Fig. 5. Comparison of analytical and simulation results with respect to the number of nodes in a cluster when $\mu^* = 0.184$.




(a) With respect to the number of nodes in a cluster.

(b) With respect to the maximum battery capacity of nodes.

Fig. 6. Comparison of the cluster lifetime with five different node scheduling methods.

energy of each node is drawn uniformly at random from the range $(0, \phi_{\max}]$, where $\phi_{\max} = 100, 300, \cdots, 900$. The results are shown in Fig. 6b. Again, both ONS and GNS outperform the rest with invGNS being the worst.

## VI. Related Work

It has been widely studied how to prolong the network lifetime and/or satisfy certain requirements for wireless sensor or IoT networks. The key challenge in such studies is to determine which nodes to activate so that the rest can switch to the sleep mode without violating given constraints. In addition, energy harvesting [6] or simultaneous missions [7] [8] have been considered when designing a node scheduling algorithm. One drawback of these works is that, it is usually assumed the requirements can be represented as simple equations in the problem formulation. Such approaches may not be realistic because some of the practical requirements are tightly coupled with the specifications of individual nodes or the underlying network. Thus, it is non-trivial to derive a practical and efficient solution to the node activation/sleeping scheduling problem under realistic requirements.

Studies on energy-efficient node scheduling are often based on the mathematical understanding of the network. For example, by using Markov chain [9] or the study on the correlation between the sensed readings [10] [11], node scheduling schemes have been developed. However, these studies either focus on the energy-efficiency of each individual node [9] or use an indirect metric when designing a network lifetime maximization problem [10] [11], instead of considering the direct relation between the nodal residual energy and the expected network lifetime. In addition, different node scheduling problems have been studied, in [12] by optimizing the transmission time but without exploiting the sleep mode, in [5] for multi-hop networks, and in [13] for coverage optimization, respectively.

## VII. Conclusion

In this paper, we studied the problem of activation scheduling of IoT devices for environment monitoring applications. We first analyzed the network of our interest by using a Markov process. From the stationary distribution of CTMC, we computed the minimum number of devices to activate to satisfy both report-accuracy and timely-update requirements. Then, we derived an optimal scheduling algorithm for device activation that maximizes the cluster lifetime by taking into consideration the nodal residual energy. We also proposed a greedy scheduling algorithm to produce the optimal performance but with a significantly lower complexity. Evaluation results show that the proposed schemes, ONS and GNS, can effectively prolong the network lifetime. The security issue in data collection and fusion is beyond the scope of this paper, and is left for future work.

## References

[1] "IEEE P802.11 Wireless LANs: Specification Framework for TGah," IEEE 802.11-11/1137r15, May 2013.
[2] T. Kim and J. M. Chang, "Enhanced power saving mechanism for large-scale 802.11ah wireless sensor networks," *IEEE Trans. on Green Commun. Netw.*, vol. 1, no. 4, pp. 516–527, Dec. 2017.
[3] C.-T. Cheng, C. K. Tse, and F. C. M. Lau, "An energy-aware scheduling scheme for wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 7, pp. 3427–3444, Sept. 2010.
[4] MathWorks. Inc., Matlab, http://www.mathworks.com.
[5] Y. Zhao and J. Wu, "On maximizing the lifetime of wireless sensor networks using virtual backbone scheduling," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 8, pp. 1528–1535, Aug. 2012.
[6] T. L. Porta, C. Petrioli, and D. Spenza, "Sensor-missiong assignment in wireless sensor networks with energy harvarsting," in Proc. of *IEEE SECON* Salt Lake City, UT, 2011, pp. 413–421.
[7] M. P. Johnson, H. Rowaihy, D. Pizzocaro, Amotz Bar-Noy, Stuart Chalmers, Thomas La Porta, and Alune Preece, "Sensor-mission assignment in constrained environments," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 11, 2010, pp. 1692–1705, Feb. 2010.
[8] H. Rowaihy, M. Johnson, A. Bar-Noy, T. Brown, and T. L. Porta, "Assigning sensors to competing missions," in Proc. of *IEEE Globecom*, New Orleans, LA, 2008, pp. 1–6.
[9] B. Yener, M. Magdon-Ismail, and F. Sivrikaya, "Joint problem of power optimal connectivity and coverage in wireless sensor networks," *Wireless Netw.*, vol. 13, no. 4, pp. 537–550, Aug. 2007.
[10] C. Liu, K. Wu, and J. Pei, "An energy-efficient data collection framework for wireless sensor networks by exploiting spatiotemporal correlation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 7, pp. 1010–1023, Jul. 2007.
[11] H. Jiang, S. Jin, and C. Wang, "Prediction or not? An energy-efficient framework for clustering-based data collection in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 6, pp. 1064–1071, Jun. 2011.
[12] Y. Yao and G. B. Ginnakis, "Energy-efficient scheduling protocols for wireless sensor networks," in Proc. of *IEEE ICC*, Seoul, Korea, 2005, pp. 2759–2763.
[13] A. Adulyasas, Z. Sun, and N. Wang, "Connected coverage optimization for sensor scheduling in wireless sensor networks," *IEEE Sensors J.*, vol. 15, no. 7, pp. 3877–3892, Jul. 2015.