Optimizing Sensor Deployment and Maintenance Costs for Large-Scale Environmental Monitoring

Xiaofan Yu, *Student Member, IEEE,* Kazim Ergun, *Student Member, IEEE,* Ludmila Cherkasova, *Member, IEEE,* and Tajana Šimunić Rosing, *Fellow, IEEE*

Abstract*—Recent advances in low-power long-range communication schemes such as LoRa have opened up new potentials in large-scale Internet-of-Things (IoT) applications, especially environmental monitoring. However, the versatile environment and the long traveling distance have imposed significant challenges to maintenance. Previous research has shown that higher temperature exponentially accelerates electronics failure rates. The maintenance cost can take as much as 80% of the total deployment expenses if not managed carefully. In this paper, we formulate a sensor deployment problem to preventively minimize maintenance costs while ensuring tolerable sensing quality and complete connectivity. We are the first to derive a maintenance cost model for IoT networks considering thermal degradation and battery depletion. To assess the spatial phenomena of interest, we adopt the *sensing quality* metric based on mutual information. While the proposed problem is non-convex, we bring up a relaxed form and solve it with SNOPT. We further apply two population-based metaheuristics, i.e. Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC) algorithm, to approximate the optimal solution. Extensive simulations are performed on two real-world datasets of the Southern California region in the US. Our metaheuristics save up to 40% of maintenance cost compared with existing greedy heuristics under the same acceptable *sensing quality*.

***Index Terms*—Environmental Monitoring; Maintenance Cost; Sensing Quality; Metaheuristic.**

I. Introduction

The emerging Low-Power Wide Area Networks (LPWAN) technologies have opened up new possibilities for large-scale Internet-of-Things (IoT) applications. LoRa, as a typical LPWAN technique designed for IoT, is able to span up to 10 km with approximately 4.5 years lifetime on a 2 Ah battery [1]. Nearly a quarter of overall thirty billion IoT devices are assumed to be connected through LPWAN [2]. One promising direction of such large-scale IoT is environmental monitoring. Compared to event monitoring that focuses on several point-of-interests, there is more potential in providing a fine-grained distribution map of the interested phenomenon across an entire region. Examples of such continuous phenomena include temperature and particle concentration, which are essential for applications like Smart City [3] and Smart Agriculture [4]. Since we are only able to deploy a finite number of sensors into a field, a sophisticated decision process is required to select the location set from a continuous space. Previous works also optimized the communication cost [5] and network lifetime [6] during sensor deployment. However, the aspect of maintenance cost is overlooked by existing literature. In this paper, in contrast to previous works, we pay major attention to optimizing the expenses in fixing hardware failures of a deployed IoT network.

Manuscript received April 17, 2020; revised June 12, 2020; accepted July 6, 2020. This work was partially supported by SRC task #2805.001, and in part by NSF under Grants #1911095, #1826967, #1730158 and #1527034.

X. Yu, K. Ergun and T. S. Rosing are with University of California at San Diego, La Jolla, CA 92093 USA (e-mail: x1yu@ucsd.edu; kergun@ucsd.edu; tajana@ucsd.edu).

L. Cherkasova is with Arm Research, San Jose, CA 95134 USA (email: Ludmila.Cherkasova@arm.com).

This article was presented in the International Conference on Hardware/Software Codesign and System Synthesis 2020 and appears as part of the ESWEEK-TCAD special issue.

Maintenance cost is playing an increasingly significant role in large-scale environmental monitoring. Previous research has shown that higher temperature exponentially accelerates electronics failures [7], [8]. Without careful management, the maintenance cost can reach up to 80% of the total expenses in deploying IoT [9]. More concretely, Cisco estimated that for every 100K devices, $3.2M/year will be spent in administrative labor and technical support due to system failures [9]. Regarding environmental monitoring, our colleagues running the HPWREN, an educational sensor network covering the Southern California region [10], shared that a vast majority of effort is spent in maintaining the network, especially after extreme weather conditions. A lot of base stations and sensors are positioned at elevated spots for good signal strength, which makes them even harder to reach. All of these efforts do not count the long geographical distance to travel to the deployed spot. Maintenance investment has become a critical bottleneck for IoT network deployment. To improve the situation, existing works have proposed to place redundant devices at critical nodes so that the network is more fault-tolerant [11]. While this idea could temporarily mitigate the maintenance burden, it does not solve the inherent problem. In this paper, we explore how to design a reliable IoT network by modeling the maintenance cost and optimizing it from the very first step of deployment.

The intuition of the maintenance cost-oriented deployment can be explained with a motivating example. Suppose the target is to deploy a sensor into a 1D space where the sensing quality and maintenance cost over the continuous space are shown in Fig. 1. Sensing quality reflects the information gain by placing the sensor, while maintenance cost is determined by the mean time to failure (MTTF) of a device, i.e. the expected time to failure. Given the cost to fix per failure, the lower the MTTF is, the higher the maintenance cost is. The detailed procedure to compute sensing quality and maintenance cost will be explained later. In the example, the sensing quality and MTTF distributions are generated in a plausible manner based

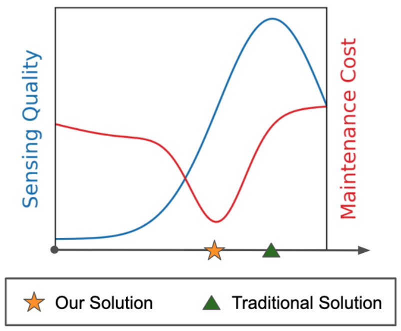


Fig. 1: Comparison of our maintenance cost-oriented sensor solution and traditional sensing quality-oriented solution.

on our observations on real-world datasets. Traditional sensor deployment solution purely searches for the best sensing quality. However, as shown in Fig. 1, the traditional solution disregards the maintenance cost which depends on the ambient temperature, the power of a device, the routing path in a network, etc. The maintenance cost of the traditional solution is expensive albeit the great sensing quality. In contrast, our solution is better as it trades sensing quality for maintenance so that the deployed sensor has much longer MTTF while could still estimate the global distribution with acceptable precision.

In this paper, we study the optimization problem of placing static sensors into a continuous 2D space for environmental monitoring. To argue the spatial informativeness of a sensing location to the global knowledge, we adopt the *sensing quality* metric in [12], which is defined as the uncertainty reduction in predicting sensing readings at unmonitored locations, given a current sensor deployment. Here the sensing variables are modeled as *Gaussian Process*. The original work of [12] selects a subset of candidate locations to place sensors. In our formulation, we improve their work by considering a continuous candidate space and possible noises in observations.

The contribution of this paper is four-fold:

- To the best of our knowledge, we are the first to develop a formal model of maintenance cost for an IoT network. We focus on permanent failures including electronics failures and battery depletion. To accurately predict the MTTF and battery depletion time, we employ state-of-the-art thermal aging, battery lifetime and power models, all incorporating the exponential temperature factor.
- We rigorously formulate a sensor deployment problem optimizing for minimum maintenance cost while satisfying acceptable sensing quality and complete connectivity. Although the problem is non-convex and nonlinear, a relaxed version is brought up and solved with a commercial optimizer.
- To approximate the optimal solution within limited time, we apply two metaheurisitcs based on Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC) Optimization using our own fitness function.
- We conduct extensive simulations in the Southern California region based on two historical datasets from PurpleAir [13]. Our results show that the proposed metaheuristics save up to 20% and 40% maintenance cost in comparison to greedy heuristics on the two datasets.

The rest of the paper is organized as follows: Section II reviews existing literature in sensor deployment and reliable IoT networks design. In Section III, we introduce our novel model for maintenance cost considering thermal factors. Section IV describes the methodology to estimate sensing quality. Integrating the models in Section III and IV together, we rigorously formulate a sensor deployment optimization problem and bring up its simplified convex form in Section V. Two metaheuristics are implemented as in Section VI and exhaustively tested in simulations based on real-world datasets in Section VII. Finally, the paper concludes in Section VIII.

II. Related Work

A. Sensor Deployment for Environmental Monitoring

Traditional research on sensor deployment started from binary event detection, which reports 0 or 1 indicating a successful detection or not. By assuming a disc-like sensing model, either binary or probabilistic, they aimed to monitor the target area with the least number of sensors [14]. Later works imposed practical constraints to the original formulation (e.g. communication [5], lifetime [6]), or transformed the problem into a multi-objective optimization [15]. However, the original event detection model cannot adapt to current environmental monitoring applications that output a continuous reading, e.g. temperature or particle concentration. In such a situation, the goal of sensor deployment shifts to placing sensors at the most informative positions.

Krause *et al.* [12], [16] first provided theoretical support for continuous-reading sensor deployment. Given pre-deployment data, they modeled sensor readings at a collection of discrete locations by *Gaussian Process* (GP). After defining *sensing quality* from mutual information, they formulated the sensor deployment problem as finding a subset from the candidate locations for best *sensing quality* and minimum communication cost. In [16], the authors provided a rigorous proof on the *NP-completeness* of the problem. They demonstrated that their polynomial-time heuristic pSPIEL is at most a constant factor worse than the optimal solution. For evaluation, a proof-of-concept experiment is conducted for indoor illumination and temperature, showing pSPIEL achieves less prediction error and communication cost than other greedy methods.

Utilizing the GP model, multiple following works studied sensor deployment for concrete applications. Wu *et al.* [17] directly applied the greedy mutual-information maximization algorithm to soil moisture monitoring. Du *et al.* [18] made an effort on deploying wind sensors at the most informative positions based on simulation results from Computational Fluid Dynamics. Recent works by Boubrima *et al.* [19] studied selecting the minimum number of grid points to ensure connectivity and sufficient monitoring accuracy for urban air monitoring. Values at non-monitored positions are estimated via interpolation. Benefiting from the linear interpolation equation, they ended up with an Integer Linear Programming problem and solved it by CPLEX.

While these works strove for both theoretical and practical progress, they present the following drawbacks: (i) Their methods required a set of discrete candidate locations as they claimed that sensors can only be installed at certain positions, e.g. lampposts. Considering the emerging technologies in durable batteries and low-power designs, this restriction can be omitted. (ii) All of the previous methodologies assumed noise-free sensors which is impractical in the real world. (iii) They failed to consider lifetime and reliability factors which are increasingly critical in large-scale deployments. In this paper, we address all of the above shortcomings. We consider sensor deployment problem in a continuous space with noisy observations. Moreover, for the first time, we construct a maintenance cost model for a complete network.

B. *Reliability-Oriented Design in IoT Networks*

As IoT networks scale up, maintenance cost will gradually replace installation cost as the dominant role in expenditure. However, only a few of existing work takes maintenance cost or reliability into account. Divergence in previous literature happens as reliability involves multiple aspects from hardware to software, from wireless link quality to environmental uncertainties, and also deployment strategies [20]. In terms of reliable sensor placement, a large number of existing works explored the *k-coverage* problem, i.e. any target needs to be covered by at least k sensors, where k is a predetermined constant [5], [6], [11]. Such a mechanism guarantees reliable event monitoring in that failure of any $k-1$ nodes will not impede successful sensing.

Apart from enhancing monitoring reliability, another group of literature focused on reliable data transmission, namely the *m-connectivity* problem [21], [22]. Bhuiyan *et al.* [21] studied the multi-objective sensor placement problem for structural health monitoring, optimizing for sensing quality, communication cost as well as network lifetime. The *m-connectivity* constraint, i.e. any edge device should be assured to have m disjoint paths to the sink, is imposed on the optimization problem. Specifically, the authors proposed a three-step sensor placement heuristic with separate stages for high-end nodes, low-end nodes, and redundant nodes. They also come up with a recovering algorithm that efficiently determines alternative routing paths when device or link faults happen.

All of the above-mentioned works contribute to a robust IoT network design from the deployment aspect. Their key idea is to insert redundancy into the network to provide better fault tolerance when any failure occurs. However, this redundancy comes with a higher cost and more effort in both installation and maintenance. Broken devices still need repair or replacement at the end, therefore increasing redundancy does not reduce maintenance cost. To the best of our knowledge, none of the previous work has approached the reliable IoT network design from limiting potential maintenance expenditure. In this paper, we aim to optimize deployment strategies which preventively minimize the estimated maintenance cost. We rigorously formulate a maintenance cost model taking state-of-the-art thermal degradation model, battery depletion model, and power model into account.

III. Maintenance Cost Model

The proposed model interprets how much effort is needed to maintain a complete IoT network. Taking hardware budget into account, we are only able to deploy m sensors in the field considering hardware installation budget. The network is uniquely determined by the *sensor deployment*, which is a set of locations $\mathcal{A} = \{a_i \mid a_i \in S, i = 1, ..., m\}$ to place sensors. All the locations belong to a convex 2D space S, which is also the area to be monitored. For ease of understanding, we summarized important symbols used in our formulation in Table I.

TABLE I: Important notations used in problem formulation.

Symbol	Meaning
S	A convex 2D deployable space
$\mathcal{A}$	A set of candidate locations to deploy sensors
$\mathcal{V}$	A set of reference locations to evaluate sensing quality
m	Number of candidate locations, i.e. $\vert\mathcal{A}\vert$
n	Number of reference locations, i.e. $\vert\mathcal{V}\vert$
R_M	The ratio of maintenance cost
c_{bat}, c_{node}	Cost for battery and node replacement
R_{MTTF}	The ratio of electronics mean-time-to-failure
R_{bat}	The ratio of battery lifetime
T_{amb}	Ambient temperature
P	Power consumption of a node
T_c	Core temperature inside the chip
T_{ref}	Standard ambient temperature of 25 °C
P_{ref}	Standard power of 165 mW
$T_{c,ref}$	Standard core temperature at T_{ref} and P_{ref}
V_{dd}	Supply voltage of battery
Cap	Initial capacity of the battery
f	Clock frequency of the SoC system
c	Location of the sink
r	Allowed communication range of sensors
$\Gamma(p)$	Set of neighbors of node p
d_{pd}	Euclidean distance between node p and q
g_{pq}	Flow quantity from node p to q
B	Communication bandwidth on the link
T	Uniform sampling interval
R	Generated data size of each sample
$F(\mathcal{A})$	Sensing quality function given $\mathcal{A}$
σ_n^2	Variances of potential sensing noises
Q	Predetermined sensing quality threshold
$Fit(\mathcal{A})$	Fitness function used in metaheurstics
P_e	Penalty to each unconnected node in $Fit(\mathcal{A})$

In general, possible failures in IoT networks include link failures, software failures and hardware failures [20]. We recognize that both link and software failures can be avoided or recovered quickly if designed properly. For example, a software bug can be fixed by updating firmware remotely. Sparse and low-rate environmental sensing applications do not impose a heavy communication burden. Thus our maintenance cost formulation focuses on permanent hardware faults, which require repair, component replacement or complete node replacement. In this paper, we only consider sensor platforms powered by single-use batteries and leave energy-harvesting devices with rechargeable batteries for future work.

The hardware crashes can be attributed to (i) power outage caused by battery depletion, (ii) electronics failures due to degradation [23]. When the battery depletes, a battery substitution is required. If the electronics break down, expert engineers are called to diagnose and a complete replacement may be performed in the worst case. Assume the cost for battery replacement at location p is $c_{bat}(p)$, while the cost for node replacement is $c_{node}(p)$, we define the expected average ratio of maintenance cost for a deployed network $\mathcal{A}$ as follows:

$$R_M(\mathcal{A}) = \sum_{p \in \mathcal{A}} \frac{c_{bat}(p)}{R_{bat}(p)} + \frac{c_{node}(p)}{R_{MTTF}(p)}, \tag{1}$$

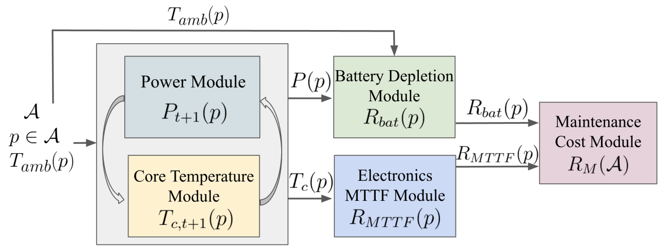


Fig. 2: Block diagram of the proposed maintenance cost model given a deployment $\mathcal{A}$ and an underlying temperature distribution T_{amb}.

where R_{bat} is the estimated ratio of battery depletion time compared to an identical battery working under the standard environment (i.e. $T_{ref} = 25$ °C and $P_{ref} = 165$ mW). Similarly, R_{MTTF} is the estimated ratio of electronics' mean time to failure in conmparison to its corresponding value in the standard environment. Since R_{bat} and R_{MTTF} are expressed in ratios, the resulted R_M is also a ratio of maintenance cost compared to the standard environment. The proportion of battery replacement cost and node replacement cost can be adjusted by the ratio of c_{bat}/c_{node}.

Fig. 2 depicts the block diagram of the proposed maintenance cost model, showing clear dependency among submodules. In the following Sections III-A-III-D, we explain the specific models used in each submodule. It should be noted that all the models in this paper aim at estimating the long-term average of a device node.

A. Electronics MTTF Module

In terms of electronics failures, previous works have thoroughly studied reliability models for microprocessors and validated these models on simulators [8], [24] as well as mobile systems [25]. The commonly studied failure mechanisms include time-dependent dielectric breakdown, negative bias temperature instability, electromigration, and thermal cycling. The temperature factor is extremely important in all of the mechanisms because the MTTF will be exponentially worse as temperature increases. In this paper, we adopt the state-of-the-art MTTF models in [26]. We notice that all of the MTTF models share a similar form on temperature dependency. We therefore extract this uniform dependency and express R_{MTTF} at core temperature T_c as a ratio to the standard core temperature $T_{c,ref}$:

$$R_{MTTF}(T_c(p)) = \exp(\frac{E_a}{kT_c(p)})/\exp(\frac{E_a}{kT_{c,ref}}), \tag{2}$$

where E_a is the activation energy, k is Boltzmann's constant. The core temperature T_c is the stabilized core temperature of a device at location p provided by the core temperature module. $T_{c,ref}$ is predetermined also by the core temperature module at the standard environment of T_{ref} and P_{ref}. Given that T_c depends on both ambient temperature and power, R_{MTTF} relies on the location and the ambient temperature distribution over the terrain, as well as the power and the routing graph.

B. Battery Depletion Module

To account the thermal effect on chemical reaction rate inside the battery, we use the T-KiBaM model proposed by Rodrigues *et al.* [7] to estimate the battery lifetime. T-KiBaM assumes two charge tanks (i.e. available charge and bound charge) to model the behavior of high-capacity lead-acid batteries. The available charge tank holds the electrical charge that can immediately supply the load, while the bound charge tank holds the secondary charge flowing towards the available charge tank. In this way, T-KiBaM successfully characterizes the non-linear *recovery effect*, i.e. the battery regains some of its "lost" capacity during idle periods [27]. The flow rate between the two tanks is regulated by their height difference and the temperature. A battery is recognized as empty when its available charge tank depletes. Let i and j denote the remained charge in available charge and bound charge tank respectively. Equation 3 depicts the differential discharging pattern of each tank, where t denotes for elapsed time. P is the average power draw from the battery computed by the power module, and V_{dd} is the supply voltage. In T-KiBaM, the reaction rate k is characterized by the Arrhenius Equation (Equation 4) from the ambient temperature T_{amb}. Parameters c, A, R, E_a in Equation 3 and Equation 4 are predefined constants for a given battery type [7].

$$\begin{cases} \frac{di}{dt} = -\frac{P}{V_{dd}} - k(1-c)i + kcj. \\ \frac{dj}{dt} = +k(1-c)i - kcj. \end{cases} \tag{3}$$

$$k(T_{amb}) = Ae^{-\frac{E_a}{RT_{amb}}}. \tag{4}$$

Suppose i_0 and j_0 are the initial capacity of each charge tank at $t = 0$. The total maximum capacity of the battery is $Cap = i_0 + j_0$. The battery lifetime L_{bat} (i.e. the instant that the available charge tank depletes) can be calculated as: $L_{bat}(P, T_{amb}) = \min t$ s.t. $i(t, P, T_{amb}) \leq 0$. The authors of [7] offered an iterative algorithm to carry out this computation. Notice that P and T_{amb} both depend on the deployed location p in our model. Our battery depletion module outputs the battery lifetime in a ratio to its standard lifetime under T_{ref} and P_{ref}:

$$R_{bat}(P(p), T_{amb}(p)) = \frac{L_{bat}(P(p), T_{amb}(p))}{L_{bat}(P_{ref}, T_{ref})}. \tag{5}$$

C. Power Module

The power consumption of a sensor determines not only its battery lifetime but also the degradation rate. Estimating power is complex since power strongly correlates to (i) sensor distribution that influences transmission distances, (ii) workload settings such as sampling rate and routing decisions, (iii) chip temperature which exponentially affects power leakage. Thus the power model should depend on the *sensor deployment* $\mathcal{A}$, the specific location p, and the core temperature T_c.

The overall power consumption of a node can be categorized as the power of the systems-on-chip (SoC) (P_{SoC}), the communication power (P_{comm}), and the power of other peripherals (P_{per}). We express the power of a node as in Equation 6 and explain the model for each piece in the following lines.

$$P(\mathcal{A}, p, T_c) = P_{SoC} + P_{comm} + P_{per}. \tag{6}$$

The power of the SoC system is composed of dynamic power p_d and leakage power p_s (which is also called static power) [28]. Dynamic power is resulted from the logic gate switching while leakage power is strongly affected by temperature. It is highly necessary to consider leakage power as it can account as much as 50% of the total power consumption using current CMOS technologies [28]. In this paper, since we mainly focus on the effect of temperature, we simplify the state-of-the-art SoC power model in [29] as:

$$P_{SoC} = p_d + p_s = aV_{dd}^2 f + V_{dd}(bT_c^2 \exp\frac{c}{T_c} + d). \tag{7}$$

Hereby, V_{dd} and f are the supply voltage and the clock frequency of the core respectively, while T_c is the core temperature in Kelvin. The constants a, b, c and d are hardware-specific parameters.

In terms of the communication power model, we modify the state-of-the-art energy model for general low-power sensor nodes in [30]. Suppose each sensor reports R Bytes of data in a uniform interval T. Without loss of generality, we use g_{pq} to denote the flow quantity from node p to q. The Euclidean distance between p and q is d_{pq}. Node p can be directly connected to q if q locates in p's disc-like binary communication range. We define a set of reachable nodes as neighbors, namely $\Gamma(p) = \{q \in S \text{ where } d_{pq} < r\}$. The communication range r is determined according to specific hardware capabilities. The average communication power of a sensor given a deployment $\mathcal{A}$ and a specific location $p \in \mathcal{A}$ can be expressed as follows:

$$P_{comm} = \frac{1}{T}\left[\sum_{q\in\Gamma(p)} P_{tx}(p,q)\frac{g_{pq}}{B} + \sum_{p\in\Gamma(q)} P_{rx}\frac{g_{qp}}{B}\right] + P_{co}. \tag{8}$$

$$P_{tx}(p,q) = p_{to} + kd_{pq}^{\alpha}. \tag{9}$$

The communication power is composed of transmission power (P_{tx}), reception power (P_{rx}), and ambient power (P_{co}) that counts for activities such as sleep, synchronization, etc. Leveraging the energy model in [30], we use constant P_{rx} while P_{tx} is characterized by the transmission distance due to the energy consumption of power amplifiers (Equation 9). Here α is the path loss exponent while k is a predetermined constant depending on channel attenuation and specific modulation techniques. p_{to} is the constant power consumed by transmission circuits. We use the typical values in [30] for P_{rx}, P_{to}, k and α. The transmission and reception time are characterized by the flow quantity g_{pq} and bandwidth B.

Finally, the peripheral power can largely vary between specific peripheral types. The peripheral can take the form of a periodic sensor, a continuously operating actuator, etc. Thus, we leave P_{per} in Equation 6 in a general form.

D. Core Temperature Module

In the above content, we explained models to estimate electronics MTTF, battery lifetime and power consumption. There is still one missing component: Equation 2 and 7 use the core temperature T_c while we only have the location-based ambient temperature $T_{amb}(p)$ as input. To convert the ambient temperature to the stabilized core temperature, we employ the latest thermal model in [31] that presents a linear iterative form within discrete-time space:

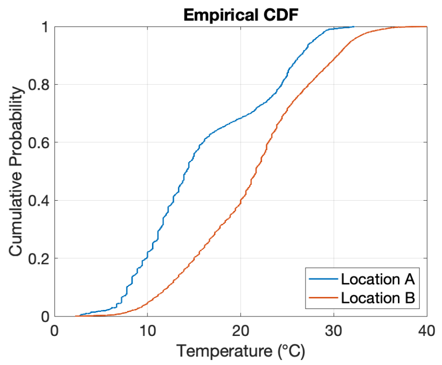


Fig. 3: The cumulative distribution of ambient temperature over one year at two different locations, which are 20 km apart. The temperature distribution data is downloaded from PurpleAir for Southern California, US [13].

$$T_c[t+1] = AT_c[t] + BP[t] + CT_{amb}[t]. \tag{10}$$

Here t is the discrete time stamp. T_c, T_{amb} and P refer to core temperature, ambient temperature, and the overall power consumption given by Equation 6 respectively. A, B and C are constant parameters obtained by fitting into experimental results. It can be seen that the core temperature T_c and the total power P depend on each other. Therefore, we implement an iterative algorithm that calls the power function and the core temperature function repeatedly until convergence, as shown in Fig. 2. The algorithm reports the stabilized value of each, which is an estimation of their long-term averages. $T_{c,ref}$ is also calculated by a similar routine using T_{ref} and P_{ref}.

E. Maintenance Cost under Temperature Variations over Time

With the model described above, we are able to estimate the maintenance costs of a given IoT network based on the power of a device $P(p)$ and the ambient temperature $T_{amb}(p)$ at location p. However, the ambient temperature distribution varies not only over space but also over time. Fig. 3 depicts the temperature distribution over one year time horizon at two locations that are 20 km apart. It can be seen that while the average temperature at location A and B is 14 °C and 22 °C, the max temperature at each location could be much higher. In 10% of the time, location B experiences a temperature higher than 30 °C. To account the temperature variations over time, we compute the expectation of R_{MTTF} and R_{bat} at location q using the integral in Equation 11. For simplicity we use P and T_{amb} in abbreviation of $P(q)$ and $T_{amb}(q)$ respectively. $p_{T_{amb}}$ is the probability associated with the temperature distribution at location q.

$$R_{MTTF}(q) = \int_{-\infty}^{\infty} R_{MTTF}\left(T_c(P, T_{amb})\right) p_{T_{amb}} dT_{amb} \tag{11a}$$

$$R_{bat}(q) = \int_{-\infty}^{\infty} R_{bat}(P, T_{amb}) p_{T_{amb}} dT_{amb} \tag{11b}$$

The output of Equation 11 are fed to the maintenance cost evaluation function depicted in Equation 1. Fixing the battery and power settings, location B in Fig. 3 results in 0.9x MTTF and 1.1x maintenance cost compared to location A. In other words, even though the average ambient temperatures are similar, the maintenance costs at this single node could be 10% different purely due to the ambient temperature variations. Referring back to the Cisco's case study, such percentile difference in maintenance costs is equivalent to \$320K/year per 100K devices by Cisco's estimates [9].

IV. Sensing Quality Evaluation

In this section, we provide the background in reasoning about sensing quality from the spatial distribution of the network. We consider deploying devices into a convex 2D space $S \subseteq \mathbb{R}^2$ for monitoring physical phenomenon. We are only able to deploy m devices in the field.

Suppose $\mathcal{A}$ is the selected deployment locations while $\mathcal{V}$ is a set of reference locations that are unmonitored. $\mathcal{X}_\mathcal{A}$ and $\mathcal{X}_\mathcal{V}$ are the random variables denoting sensor readings associated with $\mathcal{A}$ and $\mathcal{V}$. By deploying sensors at $\mathcal{A}$, the goal is to predict $\mathcal{X}_\mathcal{V}$ as accurate as possible with $\mathcal{X}_\mathcal{A}$. We use the *sensing quality* metric defined in [12] and their corresponding computation:

$$F(\mathcal{A}) = \frac{H(\mathcal{X}_\mathcal{V}) - H(\mathcal{X}_\mathcal{V} \mid \mathcal{X}_\mathcal{A})}{H(\mathcal{X}_\mathcal{V})}, \tag{12}$$

where $H(\mathcal{X}_\mathcal{V})$ is the entropy of $\mathcal{X}_\mathcal{V}$, $H(\mathcal{X}_\mathcal{V} \mid \mathcal{X}_\mathcal{A})$ is the conditional entropy of $\mathcal{X}_\mathcal{V}$ given $\mathcal{X}_\mathcal{A}$. Therefore, $H(\mathcal{X}_\mathcal{V}) - H(\mathcal{X}_\mathcal{V} \mid \mathcal{X}_\mathcal{A})$ is the mutual information between $\mathcal{X}_\mathcal{V}$ and $\mathcal{X}_\mathcal{A}$, which is the amount of information obtained about $\mathcal{X}_\mathcal{V}$ through observing $\mathcal{X}_\mathcal{A}$. It can also be interpreted as the uncertainty reduction in the predictions at $\mathcal{V}$ given the observations at $\mathcal{A}$. We divide the mutual information by $H(\mathcal{X}_\mathcal{V})$ to normalize it and define the resulting expression as the *sensing quality*, which is a real number between 0 and 1. Here, 1 refers to the best sensing quality and 0 means the worst. As a concrete example, $F(\mathcal{A}) = 1$ indicates that we can predict the readings at $\mathcal{V}$ with deployment $\mathcal{A}$ with 100% accuracy. $F(\mathcal{A}) = 0.1$ means that by deployment $\mathcal{A}$ we can reduce the uncertainty in predicting $\mathcal{X}_\mathcal{V}$ by 10% compared to its original uncertainty. The work in [12] selects a subset from discrete candidate locations to obtain the best *sensing quality* at non-selected spots. In contrary, our target is to monitor a continuous space while the candidate space is also continuous. Therefore, we employ a uniformly distributed set of locations $\mathcal{V}$ to reason the *sensing quality* of the entire region given a deployment $\mathcal{A}$. $\mathcal{V}$ is obtained by dividing S into n fine-grained grids and including the center of each grid.

Assuming each variable x_p at location $p \in S$ follows a Gaussian distribution, we are able to estimate the mean and covariance of the distribution at a non-monitored location, conditioned on the measurements at monitored locations through *Gaussian Process* (GP). Note, that for non-Gaussian phenomenon, we can split the time axis into smaller frames and consider the distribution of each frame as GP separately [18]. A GP consisting of n Gaussian-distributed variables is completely specified by a mean vector $\mathcal{M} \in \mathbb{R}^n$ and a covariance matrix $\Sigma \in \mathbb{R}^{n\times n}$. For simplicity we use $\Sigma_{VA} \in \mathbb{R}^{n\times m}$ to denote a covariance matrix between $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ and $\mathcal{A} = \{a_1, a_2, ..., a_m\}$ whose (i, j) entry is the covariance between x_{v_i} and x_{a_j}. Covariance matrices Σ_{AA}, Σ_{VV} and Σ_{AV} are defined according to similar rules. The Radial Basis Function (RBF) kernel is employed to estimate the covariance between two undeployed locations based on the distance between them (Equation 13). To better characterize the correlation between sensors at unmonitored locations, we use pre-deployment data to fit σ.

$$\Sigma_{VA_{i,j}} = \mathcal{K}(x_{v_i}, x_{a_j}) = \exp(-\frac{\|v_i - a_j\|^2}{2\sigma^2}). \tag{13}$$

Given sensor observations $\mathcal{X}_\mathcal{A}$, the value distribution of $\mathcal{X}_\mathcal{V}$ conform to a Gaussian distribution whose conditional mean $\mathcal{M}_{\mathcal{X}_\mathcal{V}|\mathcal{X}_\mathcal{A}}$ and covariance $\Sigma_{\mathcal{X}_\mathcal{V}|\mathcal{X}_\mathcal{A}}$ are given by

$$\mathcal{M}_{\mathcal{X}_\mathcal{V}|\mathcal{X}_\mathcal{A}} = \Sigma_{VA}[\Sigma_{AA} + \sigma_n^2 I]^{-1}\mathcal{M}_{\mathcal{X}_\mathcal{A}}, \tag{14a}$$

$$\Sigma_{\mathcal{X}_\mathcal{V}|\mathcal{X}_\mathcal{A}} = \Sigma_{VV} - \Sigma_{VA}[\Sigma_{AA} + \sigma_n^2 I]^{-1}\Sigma_{AV}, \tag{14b}$$

where $\mathcal{M}_{\mathcal{X}_\mathcal{A}}$ is the average sensor readings at $\mathcal{A}$. The form in Equation 14 is slightly different from [12]: we include the potential independent identically distributed (i.i.d.) Gaussian noise ϵ with variances σ_n^2 on sensing. Such adjustment better describes the practical environment where sensor readings suffer from drifting or interference. Finally, to compute the *sensing quality*, a closed form of conditional entropy can be inferred depending solely on the determinant of $\Sigma_{\mathcal{X}_\mathcal{V}|\mathcal{X}_\mathcal{A}}$:

$$H(\mathcal{X}_\mathcal{V} \mid \mathcal{X}_\mathcal{A}) = \frac{1}{2}\log((2\pi e)^n \left|\Sigma_{\mathcal{X}_\mathcal{V}|\mathcal{X}_\mathcal{A}}\right|). \tag{15}$$

V. Problem Formulation

The goal of sensor deployment is to obtain a deployment plan $\mathcal{A}$ that gives the minimum maintenance cost while ensuring acceptable sensing quality at reference locations $\mathcal{V}$ and complete connectivity. We formulate the connectivity constraint as a network graph problem. A functional sensor network should be able to deliver the data originating from each node to the sink through multi-hop routing. Benefiting from LoRa's long communication range, a large number of IoT devices can be covered by a single sink and with a simple star topology [1]. Therefore, we assume a sink is set up at location c to collect periodic updates from all deployed sensors, while the same methodology can be easily adapted to the multi-sink case. We do not consider base station (or router) placement in this paper because the position to set up base station needs careful considerations like sufficient elevation and power supply.

Employing all of the models above, we are able to rigorously formulate the sensor deployment problem as follows.

$$\min_{\mathcal{A}} \quad R_M(\mathcal{A}) \tag{16a}$$

$$\text{s.t.} \quad (1), (2), (3), (4), (5), (6), (7), (8), (9), (10)$$

$$(12), (13), (14b), (15),$$

$$F(\mathcal{A}) \geq Q \tag{16b}$$

$$\sum_{q\in\Gamma(p)} g_{pq} - \sum_{q\in\Gamma(p)} g_{qp} = R, \quad p \in \mathcal{A} \tag{16c}$$

$$\sum_{q\in\Gamma(c)} g_{qc} = mR, \quad q \in \mathcal{A} \tag{16d}$$

$$\mathcal{A} \subset S, \quad |\mathcal{A}| = m \tag{16e}$$

Equation 16b enforces the *sensing quality* to be greater than a lower bound Q. We refer to this bound as the *sensing quality* threshold. Equation 16c and 16d ensure each deployed sensor generate a data unit in the network and all sensor readings are converged to the sink.

The formulated problem is a non-convex nonlinear optimization problem. To get the lower bound of the optimum, we first derive an approximated and relaxed version of the problem that can be solved with commercial optimization solvers. Considering a single sensor, its maintenance cost model (Equation 1), electronics MTTF model (Equation 2) and power model (Equation 6, 7, 8, 9) are inherently convex. According to our observations, the ambient temperature distribution is almost always non-convex but can be approximated well with high-order polynomials in the 2D space. For the battery model (Equation 5), we fit a concave polynomial surface with parameters P and T_{amb}. To have a linear approximation for the core temperature, we use the state-space formulation in Equation 10 and find the time step t_{ss} where temperature reaches a steady-state. We unroll the list of linear equations until time step $t = t_{ss}$ and calculate the coefficients of P and T_{amb}. We break the mutual dependency between power P and core temperature T_c by assuming a constant T_c in Equation 7. Finally, we borrow the probabilistic sensing model from [32] as a simplified sensing quality model:

$$F_s(\mathcal{A}) = \sum_{a\in\mathcal{A}}\sum_{v\in\mathcal{V}} e^{-\lambda_1 d_{av}} - \sum_{a_1\in\mathcal{A}}\sum_{a_2\in\mathcal{A},a_2\neq a_1} e^{-\lambda_2 d_{a_1,a_2}}. \tag{17}$$

Even with above relaxations, jointly optimizing the locations of m sensors is still an unfeasible non-convex problem. On the other hand, the problem of optimally adding a single sensor to an existing set of deployed sensors is a simpler, well structured problem. Instead of solving the original problem, we can solve an optimization problem for the deployment of each sensor. With this approach and all the above modifications, we obtain an approximation but solvable form of the original problem named as sOPT. sOPT sequentially determines the location to place each sensor. Each step is a non-convex subproblem, but a global optimal location can be found (per subproblem) if the temperature distribution function is not too complex. We solve sOPT with SNOPT (Sparse Nonlinear OPTimizer) [33] and use its solution as a baseline for comparison to the following heuristics.

VI. Metaheuristic Design and Implementation

Swarm Intelligence has triggered the creation of lot of efficient searching heuristics in high-dimensional space. Inspired from the nature, population-based metaheuristics employ a group of individuals to search the space while each individual communicates globally about the current optimal solution. Such collective intelligence usually ends up with a sufficiently good solution for non-convex problems. Typical swarm algorithms include Particle Swarm Optimization (PSO) [34], Artificial Bee Colony (ABC) algorithm [35], and Ant Colony Optimization [36], all of which have been applied to sensor deployment problems in previous literature [37]–[39]. In our problem, the deployment solution $\mathcal{A} \in \mathbb{R}^{m\times 2}$ can be viewed as a point in a continuous high-dimensional space. Therefore we apply PSO and ABC which are well-suited for continuous-space problems, to search the solution space.

A. Fitness Function Design

Both PSO and ABC rely on a fitness function to evaluate the quality of a solution. The fitness function should accurately reflect the optimization goal and constraints. Here, we use a linearly weighted form of (i) maintenance cost, (ii) *sensing quality*, and (iii) connectivity as follows.

$$\begin{aligned} Fit(\mathcal{A}) = w_1 R_M(\mathcal{A}) + w_2 \max(Q - F(\mathcal{A}), 0) + \\ w_3 P_e \left|\textit{unconnected nodes}\right|, \end{aligned} \tag{18}$$

where w_1, w_2 and w_3 are weighting parameters for the tradeoff between three objectives, $w_1 + w_2 + w_3 = 1$. The first term in Equation 18 is the maintenance cost objective. The second term penalizes the gap if the *sensing quality* fails to satisfy the threshold. Finally, we add a large penalty P_e to each unconnected node, i.e. the nodes which locate outside of the permitted transmission range r thus are disconnected to the rest network. $|$*unconnected nodes*$|$ returns the number of unconnected nodes. Since $\mathcal{A} \in \mathbb{R}^{m\times 2}$ is a set of locations without any connectivity information, in the fitness evaluation, we attempt to construct a Minimum Spanning Tree (MST) from $\mathcal{A}$ with all connectable nodes. The generated MST is used as the network topology in maintenance cost evaluation, while the number of isolated nodes from the sink impose the corresponding amount of penalty on the fitness value. The overall objective is to minimize the fitness value.

B. PSO-based Metaheuristic

Particle Swarm Optimization is initially inspired by bird flocks searching for corns. In PSO, a group of particles are randomly placed into the space at first. The location of each particle is a potential solution fed into the fitness function. At every iteration, each particle moves towards a direction determined by its personal best and global best locations with the minimum fitness value, with some random perturbations. Specifically, suppose the location of the ith particle at iteration t is $x^{(t)} \in \mathbb{R}^{m\times 2}$ while its velocity is $v^{(t)} \in \mathbb{R}^{m\times 2}$. $p_i^{(t)}$ is the ith individual's optimal location so far, and $p_c^{(t)}$ is the global optimal location. In our implementation, we leverage the PSO with constriction factor introduced by [40]. The velocity and location of particle i for the next iteration is updated as:

$$v_i^{(t+1)} = \chi \left\{ v_i^{(t)} + r_1 c_1 (p_i^{(t)} - x_i^{(t)}) + r_2 c_2 (p_c^{(t)} - x_i(t)) \right\}, \tag{19a}$$

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}, \tag{19b}$$

where χ is the constriction factor taking a value of approximately 0.729. $r_1, r_2 \in (0, 1)$ are random numbers, while c_1, c_2 are acceleration coefficients. The complete procedure of the PSO algorithm is stated in Algorithm 1.

C. ABC-based Metaheuristic

Similar to PSO, Artificial Bee Colony Algorithm is originally brought up to imitate honey bees' foraging behaviors. ABC has three key elements:

Algorithm 1 PSO Algorithm

```
1: Initialize a population of particles with random locations and velocities in the solution space.
2: loop
3:   for all particles do
4:     Update location and velocity as Equation 19.
5:     Evaluate its current fitness as Equation 18.
6:     Compare and update the personal best of the current particle and the global best solution.
7:   end for
8:   If a criterion is met (a sufficiently good fitness or a maximum number of iterations), exit loop.
9: end loop
```

(i) Food Source: food sources are potential solutions of the desired problem, namely deployment plan $\mathcal{A}$ in our case. Each food source is associated with a "profit" obtained from the fitness function.

(ii) Employed bees: every employed bee is coupled with a food source. In each iteration, employed bees are expected to explore an adjacent random food source of the current exploiting one and return with the information about its profitability.

(iii) Unemployed bees: there are two types of unemployed bees, i.e., onlookers and scouts. Onlookers set out to an existing food source according to the information of profitability from employed bees. With roulette wheel selection [41], a more beneficial food source will be explored with higher probability. On the other hand, an employed bee transforms to a scout once its related food source is "abandoned". In other words, the food source is reinitialized if explored more than a certain amount of times without showing any progress. The scouts will randomly search new food sources in the space.

A complete flow of the ABC algorithm is described in Algorithm 2. It can be seen that, ABC introduces more randomness than PSO with the food source discard. The tuning of the exploring upper limit is critical to the performance of ABC.

Algorithm 2 ABC Algorithm

```
1: Initialize a population of bees associated with random locations in the solution space.
2: loop
3:   Move the employed bees onto their random food sources near the associated location and determine the profit.
4:   Move the onlookers onto their random food sources near the associated one and determine the profit.
5:   If a food source is abandoned, move the scouts for searching new sources.
6:   Update the best solution so far.
7:   If a criterion is met (a sufficiently good fitness or a maximum number of iterations), exit loop.
8: end loop
```

TABLE II: Parameter settings in simulation.

Param.	Value	Param.	Value	Param.	Value
r	10 km	B	1 kbps	R	100 B
σ_n^2	10	c_{bat}	10	c_{node}	100
V_{dd}	3.3 V	Cap	2 Ah	f	300 MHz
P_{to}	520 mW	k	10^{-4}	α	3.5
P_{rx}	200 mW	T_{ref}	25 °C	P_{ref}	165 mW

VII. Simulation on Real-World Dataset

In this section, we present and discuss the simulation results based on two large-scale real-world datasets.

A. Experiment Setup

We implement our maintenance model and sensor deployment approach in MATLAB R2020a[1]. Simulations are performed on a Linux desktop with Intel Core i7-8700 CPU at 3.2 GHz and 16 GB RAM. To estimate the underlying distribution of the phenomena, we use the environmental monitoring history from PurpleAir [13] as the pre-deployment data in the simulations. The dataset contains sensor readings of temperature, humidity, and air particle (i.e. pm1, pm2.5, pm10) concentrations every 10 minutes. We simulate all deployment methods on a small and a large region in Southern California, US. For the small region of 30 km $\times$ 50 km, the pre-deployment data constitutes of 27 sensors with reading history from January 1, 2019 to February 20, 2020. In terms of the larger region of 60 km $\times$ 100 km, we use the data of 264 sensors from January 1, 2019 to April 1, 2020.

Without loss of generality, we assume the sink locates at the center of the target region. For peripheral, we consider a periodic sensor with the peripheral power characterized as follows:

$$P_{per} = p_{sensor} t_{sen} / T, \tag{20}$$

where $p_{sensor} = 200$ mW is the active power draw of the sensor peripheral, $t_{sen} = 300$ ms is the delay to update data samples and $T = 10$ s is the sampling interval. We summarize the detailed parameter settings of our maintenance model in Table II. For the fitness function, we set $w_1 = 0.5, w_2 = 0.4, w_3 = 0.1, P_e = 100$.

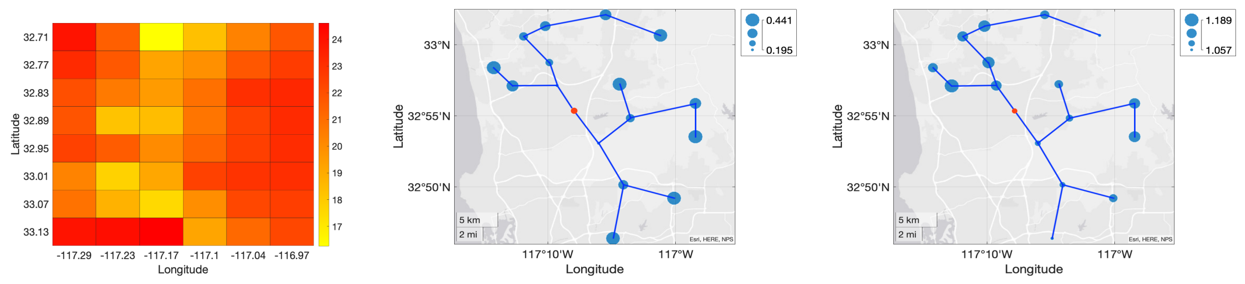
B. Baseline Setup

For comparison, we employ the following baselines and compare their output of maintenance cost with the same fixed *sensing quality* threshold:

- **Information-Driven Sensor Querying (IDSQ) [42]**: Starting from nodes adjacent to the sink, IDSQ greedily selects one reachable location from the candidate set $\mathcal{V}$ in each iteration. The goal is to maximize a weighted form of *sensing quality* and maintenance cost gain. It terminates once the pre-determined *sensing quality* threshold is satisfied. Equation 21 describes selecting location p_j in iteration j where $p_{prev} = \{p_1, ..., p_{j-1}\}$ is the existing set generated from previous selections. We set $\alpha = 0.6$.

$$\begin{aligned} p_j = \arg\max_{p} \alpha \left[F(\{p_{prev}, p\}) - F(p_{prev})\right] - \\ (1-\alpha)\left[R_M(\{p_{prev}, p\}) - R_M(p_{prev})\right]. \end{aligned} \tag{21}$$

[1]The source code is available at https://github.com/Orienfish/AQI-deploy.



(a) Prediction of ambient temperature (°C) over the region.

(b) Deployment solution provided by PSO with R_{bat} of each node represented by the radius.

(c) Deployment solution provided by PSO with R_{MTTF} of each node represented by the radius.

Fig. 4: Visualization of simulation on the small region. Red node is the sink and blue nodes are sensors. The battery lifetime and electronics MTTF values are ratios to their corresponding values in the standard environment.

- **Padded Sensor Placements at Informative and cost-Effective Locations (pSPIEL) [12]**: pSPIEL uses padded decomposition to divide candidate locations into clusters and greedily selects a subset from each cluster. The greedy rule is reaching minimum communication cost after satisfying the *sensing quality* threshold. We integrate pSPIEL into our simulation based on the toolbox offered by the original authors [43]. The candidate set is configured as $\mathcal{V}$, and the communication cost $D(p,q)$ between any two node p,q is defined as follows:

$$D(p,q) = d_{pq} + [d_{pq} < r]P_e, \tag{22}$$

where $[cond]$ gives 1 when the inner condition $cond$ is met, otherwise 0.
- **sOPT**: Given the sink node, sOPT sequentially adds a single sensor in each iteration. Each sensor is placed greedily to minimize its own maintenance cost while satisfying the *sensing quality* requirement. The solution at each iteration is calculated by SNOPT [33] solving the relaxed form of the problem described in Section V. SNOPT can handle well-structured non-linear non-convex problems as long as the function is smooth. However, due to the complexity in searching for the optimal solution, it cannot converge for very large problems, so we could only experiment with sOPT in the small-scale simulation.

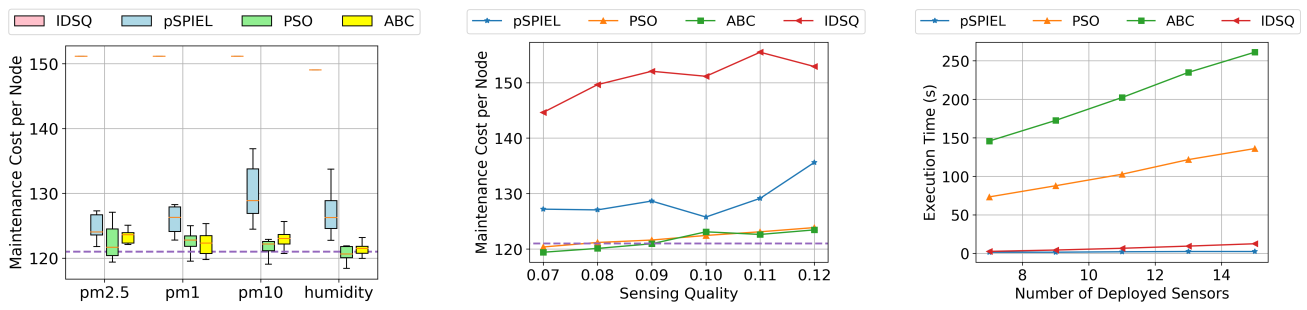
C. Simulation Results on a Small Region

Visualization. On the small-scale region of 30 km $\times$ 50 km, we intend to place 16 sensors. Reference locations are set to the center of 6×8 grids, with the predicted average temperature distribution by GP at each of them shown in Fig. 4a. Notice, that while Fig. 4a shows the average value at each location, our maintenance model takes the whole temperature distribution over time as input (see explanations in Section III-E). As a visualization, we present the deployment solution of PSO and the predicted R_{bat} and R_{MTTF} of each node in Fig. 4b and 4c. We omit the visualization of ABC's deployment solution as it presents similar patterns. It can be seen that the intersection nodes end up with shorter battery lifetime while the locations with higher average temperature generally have shorter electronics MTTF. As expected, the generated deployment solution tends to avoid the zones with harsher temperature while carefully spreading over the region.

Monitoring various phenomenons. We fix the *sensing quality* threshold to 0.1 and run each of IDSQ, pSPIEL, PSO, ABC and sOPT for 10 times on various phenomenon's dataset. Both PSO and ABC employ 20 individuals searching for 30 iterations. PSO and ABC are set to placed 16 sensors. The maintenance cost per node of resulted deployments are shown in Fig. 5a. We compare the maintenance cost per node because IDSQ and pSPIEL do not have a preset sensor counts, terminating once the threshold is reached. It can be observed that pSPIEL, PSO and ABC's solutions are fluctuated since they all rely on random initialization. IDSQ's solution is fixed in each case but it performs the worst. This is because IDSQ greedily selects adjacent locations without considering the global distribution of *sensing quality* and temperature. pSPIEL discovers much better solutions as it uses padded clustering to ensure the deployment is well-spread. While IDSQ and pSPIEL select from the finite grid locations, the proposed metaheuristics PSO and ABC explore in the entire continuous space, providing closer or even surpassing performance compared to the relaxed lower boundary given by sOPT.

The detailed improvements normalized to IDSQ's solution regarding maintenance cost, battery depletion time and electronics MTTF are summarized in Table III. For battery and electronics lifetime, we report both the average and minimum (i.e. the time when the first device fails). The improvements on electronics MTTF are relatively not significant because the temperature is mild in the target region. However, the pre-deployment data at extreme habitats such as desert is rare. If testing with a more comprehensive dataset covering a variety of environments, it is reasonable to expect more maintenance savings with our proposed methodology. Even though, our metaheuristics saves up to 20% of the maintenance cost compared to IDSQ, which transforms to \$0.64M/year per 100K devices according to Cisco's estimation [9].

Trade-offs between *sensing quality* and maintenance cost. Exploring the trade-offs between *sensing quality* and maintenance cost is a common issue in practical deployment. Using only the pm2.5 dataset, we examine the maintenance cost given by each algorithm at various *sensing quality* threshold. As depicted in Fig. 5b, the resulted maintenance cost per node increases monotonically as the threshold grows. This is reasonable as the higher the sensing demands, the wider area to cover, thus the more expenses in maintenance. Both PSO and ABC explore better solutions than pSPIEL and IDSQ, while ABC performs slightly better and more stably than PSO. ABC also converges faster in our observation.



(a) Maintenance cost in ratio for monitoring various phenomenons. The purple dashed line depicts the solution of sOPT as a higher boundary of the optimum.

(b) Maintenance cost in ratio to *sensing quality* threshold. The purple dashed line depicts the solution of sOPT as a higher boundary of the optimum.

(c) Execution time to the number of deployed nodes.

Fig. 5: Simulation results on the small region.

TABLE III: Average maintenance cost, battery depletion time and electronics MTTF improvements normalized to IDSQ's solution on the small region.

Metric	IDSQ	pSPIEL	PSO	ABC
Average Maintenance Cost	1.0	0.84	0.81	**0.80**
Average Battery Depletion Time	1.0	1.51	1.63	**1.64**
Minimum Battery Depletion Time	1.0	1.45	**1.75**	1.66
Average Electronics MTTF	1.0	1.02	**1.03**	**1.03**
Minimum Electronics MTTF	1.0	1.02	**1.04**	**1.04**

TABLE IV: Average maintenance cost, battery depletion time and electronics MTTF improvements normalized to IDSQ's solution on the large region.

Metric	IDSQ	pSPIEL	PSO	ABC
Average Maintenance Cost	1.0	0.68	0.64	**0.60**
Average Battery Depletion Time	1.0	0.99	1.04	**1.06**
Minimum Battery Depletion Time	1.0	1.69	2.29	**2.69**
Average Electronics MTTF	1.0	1.08	1.09	**1.10**
Minimum Electronics MTTF	1.0	2.61	2.68	**2.80**

Execution time. We report the execution time of all heuristics, when deploying various number of sensors in Fig. 5c. Both IDSQ and pSPIEL take less than 20 seconds to finish. We remind the reader that since IDSQ and pSPIEL select a subset from a finite candidate location set, their search space is much smaller than PSO and ABC. Both heuristics are searching in continuous space, ABC executes two times longer compared to PSO. This is because each individual in ABC is employed for two searching trials in one iteration, first as an employed bee and then as an unemployed bee. More efforts in searching bring better and more stable performance to ABC.

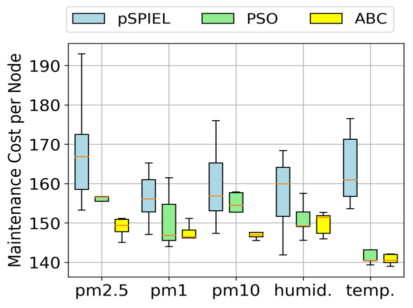
D. Simulation Results on a Large Region

In order to test our methodology on large-scale monitoring applications, we select another 60 km $\times$ 100 km region to hold 54 sensors. The selected region covers more versatile environments including urban and natural areas with more fluctuated temperature, humidity and air particle levels. The *sensing quality* is estimated with 21 $\times$ 10 grid-like reference locations. We experiment and discuss the performance of IDSQ, pSPIEL, PSO and ABC on the large region. sOPT is not evaluated due to its exponentially increasing complexity in finding the optimal solution for larger problems.

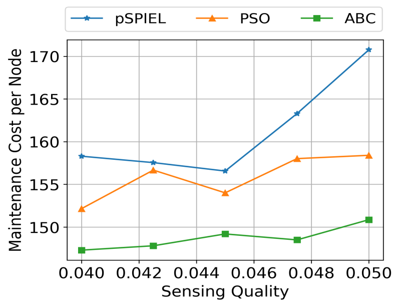
Monitoring Various Phenomenons. We first evaluate our methods on various phenomenons with fixed *sensing quality* threshold at 0.05. Due to the complexity of the larger dataset, we execute IDSQ and pSPIEL for 10 times while test PSO and ABC for 5 times. Both PSO and ABC employ 10 individuals searching for 40 iterations. The resulted maintenance cost per node are reported in Fig. 6a, where we exclude the result of IDSQ as its maintenance costs are too much to be depicted in the same scale. Compared with the small region, the maintenance-cost-per-node of all algorithms increase. Observing the average maintenance cost, PSO and ABC perform better than pSPIEL in general. ABC remains remarkable in both average and stability. In contrast to the simulation results on the small region, Fig. 6a demonstrates the strong dependability on phenomenons in large-scale deployment. The performance gap between PSO, ABC and pSPIEL is larger for temperature monitoring. The underlying distribution of phenomenons can have great impacts on the *sensing quality* and lead to largely varied maintenance cost.

The detailed ratios to IDSQ's solution on the large simulation are summarized in Table IV. It can be observed that ABC scales better than the rest algorithms on a larger deployment with versatile environments. Aside from a greater improvements across all of the metrics compared to the small simulated case, it particularly shows significant advances on minimum battery depletion time and electronics MTTF. Such outcome suggests that our proposed methodology is able to enhance the reliability of critical nodes at intersections, while existing greedy heuristics do not take this factor into account. Overall, the ABC-based heuristics saves 40% of maintenance cost compared to IDSQ and 12% compared to pSPIEL. It is equivalent to \$1.28M/year and \$384K/year for every 100K devices in Cisco's estimation [9].

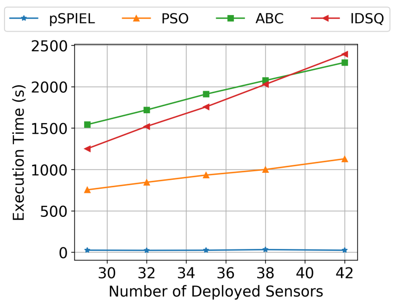
Trade-offs between *sensing quality* and maintenance cost. We repeat the same experiments on pm2.5 dataset but with a varying *sensing quality* threshold from 0.04 to 0.05. Due to the increase in scale, the possible *sensing quality* we can reach by placing finite sensors becomes lower. The resulted maintenance cost per node of pSPIEL, PSO and ABC are summarized in Fig. 6b. Again we omit IDSQ's result as it significantly exceeds the scale of the rest three algorithms. As shown in Fig. 6b, ABC presents the lowest maintenance cost and the most stable outcome across all test cases. The maintenance per node given by PSO and ABC is averagely 9% and 4% better than pSPIEL at all *sensing quality* thresholds.



(a) Maintenance cost in ratio for monitoring various phenomenons.



(b) Maintenance cost in ratio to *sensing quality* threshold.



(c) Execution time to the number of deployed nodes.

Fig. 6: Simulation results on the large region.

Execution time. The time consumption of all heuristics are reported in Fig. 6c when deploying 29, 32, 35, 38 and 42 nodes. All algorithms except pSPIEL require significantly longer time to compute the deployment on the large region. pSPIEL also shows an increase in execution time but it still lasts no more than 30 seconds. Similar as the case on the small region, ABC takes 2x longer in comparison to PSO due to more searching trials. IDSQ, differently, explodes up in execution time and takes longer than ABC in the worst case. This reflects the poor scalability of IDSQ in large-scale problems. We recognize that the long execution time of PSO and ABC come from the complexities in calculating the maintenance cost. In the future work, we will look into techniques to accelerate maintenance cost evaluation.

VIII. Conclusion

In this paper, we rigorously formulate a large-scale sensor deployment problem with the objective of minimum maintenance cost in fixing hardware failures. We, for the first time, propose a quantitative maintenance cost models considering electronics degradation and battery depletion, all incorporating the exponential temperature factor. In the meantime, we integrate the *sensing quality* metric to evaluate the spatial informativeness of the deployment with statistical analysis of pre-deployment data. While the formulated optimization problem is non-convex and nonlinear, we apply two metaheuristics to efficiently search in the high-dimensional solution space using our own fitness function. The performance of the metaheuristics are compared with existing greedy heuristics and a sequential optimal solution. Evaluation results on two real-world environmental-monitoring datasets show that our metaheuristics can save up to 40% of the maintenance cost in comparison to previous greedy heuristics.

References

[1] J. C. Liando, A. Gamage, A. W. Tengourtius, and M. Li, "Known and unknown facts of lora: Experiences from a large-scale measurement study," *ACM Transactions on Sensor Networks (TOSN)*, vol. 15, no. 2, pp. 1–35, 2019.

[2] U. Noreen, A. Bounceur, and L. Clavier, "A study of lora low power and wide area network technology," in *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*. IEEE, 2017, pp. 1–6.

[3] M. Xie, Y. Bai, Z. Hu, and C. Shen, "Weight-aware sensor deployment in wireless sensor networks for smart cities," *Wireless Communications and Mobile Computing*, 2018.

[4] O. Kaiwartya, A. H. Abdullah, Y. Cao, R. S. Raw, S. Kumar, D. K. Lobiyal, I. F. Isnin, X. Liu, and R. R. Shah, "T-mqm: Testbed-based multi-metric quality measurement of sensor deployment for precision agriculture a case study," *IEEE Sensors Journal*, vol. 16, no. 23, pp. 8649–8664, 2016.

[5] M. Rebai, H. Snoussi, F. Hnaien, L. Khoukhi *et al.*, "Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks," *Computers & Operations Research*, vol. 59, pp. 11–21, 2015.

[6] S. Mini, S. K. Udgata, and S. L. Sabat, "Sensor deployment and scheduling for target coverage problem in wireless sensor networks," *IEEE sensors journal*, vol. 14, no. 3, pp. 636–644, 2013.

[7] L. M. Rodrigues, C. Montez, R. Moraes, P. Portugal, and F. Vasques, "A temperature-dependent battery model for wireless sensor networks," *Sensors*, vol. 17, no. 2, p. 422, 2017.

[8] C. Zhuo, D. Sylvester, and D. Blaauw, "Process variation and temperature-aware reliability management," in *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*. IEEE, 2010, pp. 580–585.

[9] Cisco Jasper, "The hidden costs of delivering iiot services: Industrial monitoring & heavy equipment," Apr. 2016. [Online]. Available: https://www.cisco.com/c/dam/m/en_ca/never-better/manufacture/pdfs/hidden-costs-of-delivering-iiot-services-white-paper.pdf

[10] "High Performance Wireless Research & Education Network (HPWREN)." [Online]. Available: http://hpwren.ucsd.edu/

[11] M. Elhoseny, A. Tharwat, X. Yuan, and A. E. Hassanien, "Optimizing k-coverage of mobile wsns," *Expert Systems with Applications*, vol. 92, pp. 142–153, 2018.

[12] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, "Robust sensor placements at informative and communication-efficient locations," *ACM Transactions on Sensor Networks (TOSN)*, vol. 7, no. 4, pp. 1–33, 2011.

[13] PurpleAir LLC, "Purpleair: Real time air quality monitoring." [Online]. Available: https://www2.purpleair.com/

[14] F. Alduraibi, N. Lasla, and M. Younis, "Coverage-based node placement optimization in wireless sensor network with linear topology," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.

[15] S. Sengupta, S. Das, M. Nasir, and B. K. Panigrahi, "Multi-objective node deployment in wsns: In search of an optimal trade-off among coverage, lifetime, energy consumption, and connectivity," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 405–416, 2013.

[16] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, no. Feb, pp. 235–284, 2008.

[17] X. Wu, M. Liu, and Y. Wu, "In-situ soil moisture sensing: Optimal sensor placement and field estimation," *ACM Transactions on Sensor Networks (TOSN)*, vol. 8, no. 4, pp. 1–30, 2012.

[18] W. Du, Z. Xing, M. Li, B. He, L. H. C. Chua, and H. Miao, "Sensor placement and measurement of wind for water quality studies in urban reservoirs," *ACM Transactions on Sensor Networks (TOSN)*, vol. 11, no. 3, pp. 1–27, 2015.

[19] A. Boubrima, W. Bechkit, and H. Rivano, "On the optimization of wsn deployment for sensing physical phenomena: Applications to urban air pollution monitoring," in *Mission-Oriented Sensor Networks and Systems: Art and Science*. Springer, 2019, pp. 99–145.

[20] L. Venkatesan, S. Shanmugavel, C. Subramaniam *et al.*, "A survey on

modeling and enhancing reliability of wireless sensor network," *Wireless Sensor Network*, vol. 5, no. 03, p. 41, 2013.

[21] M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu, "Sensor placement with multiple objectives for structural health monitoring," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 4, pp. 1–45, 2014.

[22] S. Harizan and P. Kuila, "Nature-inspired algorithms for k-coverage and m-connectivity problems in wireless sensor networks," in *Design Frameworks for Wireless Networks*. Springer, 2020, pp. 281–301.

[23] I. Banerjee, P. Chanak, H. Rahaman, and T. Samanta, "Effective fault detection and routing scheme for wireless sensor networks," *Computers & Electrical Engineering*, vol. 40, no. 2, pp. 291–306, 2014.

[24] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The case for lifetime reliability-aware microprocessors," *ACM SIGARCH Computer Architecture News*, vol. 32, no. 2, p. 276, 2004.

[25] P. Mercati, F. Paterna, A. Bartolini, L. Benini, and T. Š. Rosing, "Warm: Workload-aware reliability management in linux/android," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 9, pp. 1557–1570, 2016.

[26] T. S. Rosing, K. Mihic, and G. De Micheli, "Power and reliability management of socs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 4, pp. 391–403, 2007.

[27] M. R. Jongerden and B. R. Haverkort, "Which battery model to use?" *IET software*, vol. 3, no. 6, pp. 445–457, 2009.

[28] H. Wang, L. Hu, X. Guo, Y. Nie, and H. Tang, "Compact piecewise linear model based temperature control of multi-core systems considering leakage power," *IEEE Transactions on Industrial Informatics*, 2019.

[29] Y. Liu, R. P. Dick, L. Shang, and H. Yang, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *2007 Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2007, pp. 1–6.

[30] M. Abo-Zahhad, M. Farrag, A. Ali, and O. Amin, "An energy consumption model for wireless sensor networks," in *5th International Conference on Energy Aware Computing Systems & Applications*. IEEE, 2015, pp. 1–4.

[31] F. Beneventi, A. Bartolini, A. Tilli, and L. Benini, "An effective gray-box identification procedure for multicore thermal modeling," *IEEE Transactions on Computers*, vol. 63, no. 5, pp. 1097–1110, 2012.

[32] A. Hossain, P. K. Biswas, and S. Chakrabarti, "Sensing models and its impact on network coverage in wireless sensor network," in *2008 IEEE Region 10 and the Third international Conference on Industrial and Information Systems*. IEEE, 2008, pp. 1–5.

[33] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Rev.*, vol. 47, pp. 99–131, 2005.

[34] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[35] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of global optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[36] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[37] Q. Ni, H. Du, Q. Pan, C. Cao, and Y. Zhai, "An improved dynamic deployment method for wireless sensor network based on multi-swarm particle swarm optimization," *Natural Computing*, vol. 16, no. 1, pp. 5–13, 2017.

[38] S. Mini, S. K. Udgata, and S. L. Sabat, "Artificial bee colony based sensor deployment algorithm for target coverage problem in 3-d terrain," in *International conference on distributed computing and internet technology*. Springer, 2011, pp. 313–324.

[39] X. Liu and D. He, "Ant colony optimization with greedy migration mechanism for node deployment in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 39, pp. 310–318, 2014.

[40] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[41] A. Lipowski and D. Lipowska, "Roulette-wheel selection via stochastic acceptance," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 6, pp. 2193–2196, 2012.

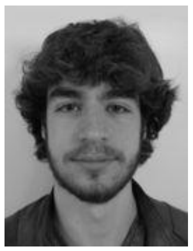
[42] F. Zhao, L. J. Guibas, and L. Guibas, *Wireless sensor networks: an information processing approach*. Morgan Kaufmann, 2004.

[43] A. Krause, "Sfo: A toolbox for submodular function optimization," *Journal of Machine Learning Research*, vol. 11, no. Mar, pp. 1141–1144, 2010.

ACKNOWLEDGMENT

This work was partially supported by SRC task #2805.001, and in part by NSF under Grants #1911095, #1826967, #1730158 and #1527034.

Xiaofan Yu (S'18) received the B.Sc. degree in electronics engineering from Peking University, Beijing, China, in 2018. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA, USA.
She is a member of the System Energy Efficiency Laboratory, University of California at San Diego. Her current research interests include reliability-driven design and implementation of IoT networks.



Kazim Ergun (S'20) received the B.Sc. degree in electrical and electronics engineering as a valedictorian from Middle East Technical University, Turkey, in 2017. He is currently working toward the Ph.D. degree with the Department of Electrical and Computer Engineering, University of California San Diego, California, USA.
He is a member of the System Energy Efficiency Laboratory where he works on optimization and control for the Internet of Things, with the focus on reliability and energy efficiency. His research interests include a wide variety of topics covering optimization and control for efficient IoT networks, wireless communications, and machine learning.

Lucy Cherkasova is a principal research scientist at Arm Research, San Jose, USA. She is focusing on the end-to-end performance, automation, reliability, and cost-effectiveness of the enterprise and IoT systems. Before that for 20+ years she was a principal scientist at Hewlett Packard Labs and led to success multiple R&D projects, with prototypes, algorithms, or features implemented in HP products. Her current research interests are in developing quantitative methods for the analysis, design, and management of distributed systems (such as emerging systems for Smart environments, Big Data processing, and next generation data centers). She is the ACM Distinguished Scientist and is recognized by multiple Certificates of Appreciation from Usenix and IEEE Computer Society.

Tajana Šimunić Rosing (F'05) received the M.S. degree in engineering management concurrently with the Ph.D. degree from Stanford University, Stanford, CA, USA, in 2001 with the Ph.D. topic "Dynamic Management of Power Consumption."
She is a Professor, a Holder of the Fratamico Endowed Chair, and the Director of System Energy Efficiency Laboratory, University of California at San Diego, La Jolla, CA, USA. From 1998 to 2005, she was a full-time Research Scientist with HP Labs, Palo Alto, CA, USA, while also leading research efforts with Stanford University, Stanford, CA, USA. She was a Senior Design Engineer with Altera Corporation, San Jose, CA, USA. She is leading a number of projects, including efforts funded by DARPA/SRC JUMP CRISP program with focus on design of accelerators for analysis of big data, DARPA and NSF funded projects on hyperdimensional computing and SRC funded project on IoT system reliability and maintainability. Her current research interests include energy efficient computing, cyber-physical, and distributed systems.