sensors

Article

Energy-Efficient Adaptive Sensing Scheduling in Wireless Sensor Networks Using Fibonacci Tree Optimization Algorithm

Liangshun Wu and Hengjin Cai *

School of Computer Science, Wuhan University, Wuhan 430072, China; wuliangshun@whu.edu.cn
* Correspondence: hjcai@whu.edu.cn; Tel.: +86-180-6205-8525

Abstract: Wireless sensor networks are appealing, largely because they do not need wired infrastructure, but it is precisely this feature that renders them energy-constrained. The duty cycle scheduling is perceived as a contributor to the energy efficiency of sensing. This paper developed a novel paradigm for modeling wireless sensor networks; in this context, an adaptive sensing scheduling strategy is proposed depending on event occurrence behavior, and the scheduling problem is framed as an optimization problem. The optimization objectives include reducing energy depletion and optimizing detection accuracy. We determine the explicit form of the objective function by numerical fitting and found that the objective function aggregated by the fitting functions is a bivariate multimodal function that favors the Fibonacci tree optimization algorithm. Then, with the optimal parameters optimized by the Fibonacci tree optimization algorithm, the scheduling scheme can be easily deployed, and it behaves consistently in the coming hours. The proposed "Fibonacci Tree Optimization Strategy" ("FTOS") outperforms lightweight deployment-aware scheduling (LDAS), balanced-energy scheduling (BS), distributed self-spreading algorithm (DSS) and probing environment and collaborating adaptive sleeping (PECAS) in achieving the aforementioned scheduling objectives. The Fibonacci tree optimization algorithm has attained a better optimistic effect than the artificial bee colony (ABC) algorithm, differential evolution (DE) algorithm, genetic algorithm (GA) algorithm, particle swarm optimization (PSO) algorithm, and comprehensive learning particle swarm optimization (CLPSO) algorithm in multiple runs.

Keywords: sensing; energy-saving; duty cycles; Fibonacci tree optimization

Citation: Wu, L.; Cai, H. Energy-Efficient Adaptive Sensing Scheduling in Wireless Sensor Networks Using Fibonacci Tree Optimization Algorithm. *Sensors* **2021**, *21*, 5002. https://doi.org/10.3390/s21155002

Academic Editor: Jaime Lloret

Received: 31 May 2021
Accepted: 15 July 2021
Published: 23 July 2021

1. Introduction

Wireless Sensor Network (WSN) is a network of thousands of low-cost miniature devices capable of processing, communicating wirelessly, and sensing, which runs on a limited battery. Since we typically expect WSN sensors to last from several months to one year without recharging, energy efficiency is essential. Much previous work has referred to energy-efficient communication (media access control and routing) in WSN ([1–4], for example); energy-efficient sensing, however, has received little attention. Results of recent measurements indicate that sensing uses comparable amounts of energy as wireless communication [5]. Additionally, sensing frequency is greater than communication frequency—for example, a sensor that monitors residential fire in forests only triggers an alarm when temperatures exceed a threshold. Therefore, it is prime time we called for efforts to study energy-saving strategies of sensing.

Dynamic events are not easily captured as they come and go and can only be observed by continuous monitoring. Energy efficiency is a serious concern if sensors are always ON. Programming sensors' work/sleep cycles (or duty cycles) is seen as a helpful response measure. When the node is in SLEEP mode, only the low-power timer remains active should it be necessary to wake it up. Hence, the energy consumed during the sleep duration is only a tiny fraction of that in the working duration.

While many studies have investigated sensing scheduling problems, they differ in their design assumptions and objectives because of the multifarious business needs of applications. Based on the belief of building up mathematical foundations and establishing synergy between detection accuracy and energy depletion (as the central theme of this paper), we suggest that the main contributions of this paper are:

- A novel system model of wireless sensor network is introduced;
- An adaptive sensing scheduling strategy based on event occurrence behavior is proposed;
- The scheduling problem is formulated as an optimization problem and is solved by a Fibonacci tree optimization algorithm.

The rest of the paper proceeds as follows: Section 2 reviews related work. Section 3 presents the system model. Section 4 introduces the Fibonacci tree optimization algorithm. Section 5 explains why the Fibonacci tree optimization algorithm is selected for parameter optimization and examines the advantages of the proposed FTOS strategy. Section 6 implements experiments. Section 7 discusses the deficiencies and unresolved issues. Finally, Section 8 draws the conclusion.

2. Related Work

2.1. Sensing Scheduling Strategies

The research on energy-saving sensing scheduling strategies involves two levels. The first level is adaptive duty cycle scheduling, concerning active time, nap time and the idle listening period. The goal is to best model the event arrival patterns. The second level, collaborative sensing, considers the sensing coverage: how numerous sensor nodes (not just a single node) with different spatiotemporal coordinates in a public area can cooperate to achieve adequate coverage, which is conducive to global optimization of energy consumption.

We know that sensors consume the most energy when they are on duty and the least when they are sleeping. As a result, almost all scheduling strategies, such as ELECTION [6] and additive increase/multiplicative decrease (AIDM) [7], make full use of the energy-saving characteristic of the sleep mode. DANCE [8] improves AIDM [7] by incorporating the behavior of neighboring nodes into consideration scope. In DANCE, the sensor abandons its task if its neighbor has already done it, thus down-scaling the energy inefficiency. The authors argue that the data sampling rate affects the computing and communication load on the central server. Controlling the sampling rate within a specific range (if it exceeds, then the central server and nodes renegotiate a range) through Kalman filtering is believed to be capable of tackling this problem [9].

Jothiraj et al. [10] recommend fine-tuning the sensing frequency of each sensor dynamically. As the sensory readings increase, the detection accuracy improves. The challenge, however, lies in modeling the actual world. Maintaining the greatest possible coverage is a primary goal when designing a sensor network. Chen et al. [11] defined a sensing coverage metric that measures a wireless sensor network's QoS (quality of service). Furthermore, a polynomial-time complexity optimization algorithm using graphing theory and computational geometry is proposed to achieve optimum coverage [12] broadens the work in [11] by enhancing the algorithm. The first work that considers both energy consumption and sensing coverage was completed by [13], which introduces an energy-efficient surveillance system (ESS). This was followed by lightweight deployment-aware scheduling (LDAS) [14,15], probing environment and adaptive sensing (PEAS) [16], probing environment and collaborating adaptive sleeping (PECAS) [17], and randomized independent scheduling (RIS) [18], and so forth. These studies make the following two common assumptions: (1) each sensor is power-constrained, and (2) the network is expected to run for a long time. Other assumptions include:

- Network Structure. The network structure can be flat or hierarchical.
- Sensor Placement. The sensing coverage is usually affected by how sensors are initially placed. In most cases, the sensors follow a random distribution or two-dimensional (2-D) Poisson distribution [12,19].
- Sensing Area: The sensing area can be 2-D circular or 3-D spherical.

- Time Synchronization. The sensors are synchronized in time, and can be woken up simultaneously for the next scheduling round.
- Failure Model. Almost all the studies assume that sensors fail when energy is exhausted.
- Sensor Mobility. Most studies have assumed that the sensor is immobile.
- Location Information. These studies typically associate location information with whether, or how much, a sensor's sensing area overlaps with its neighbor's.
- Distance Information. The distance information can be inferred from the location information.

Randomized independent scheduling (RIS) [18] can extend the sensor lifetime and obtain an asymptotic k-coverage, and, it is simple. RIS does not need to provide location information or distance information, nor does it need adjustable transmission range and mobility. However, it is based on strict distribution assumptions—for example, the sensor is Poisson/uniform/grid distributed, the sensing range follows a uniform distribution and the network is flat and two-dimensional.

Lightweight deployment-aware scheduling (LDAS) [15], probing environment and adaptive sensing (PEAS) [16], probing environment and collaborating adaptive sleeping (PECAS) [17], and so forth, make relatively loose assumptions. LDAS assumes that the sensor nodes do not own a positioning device, such as GPS, thus forces the sensors achieving the desired coverage by static sensing. PEAS consists of two mechanisms: sensing and adaptive sleep scheduling, and requires high sensor density. PECAS is an updated version of the PEAS. It posts its remaining/available hours in response to neighbors to avoid misperceptions. This leads to an increase in communication energy depletion. Balanced-energy scheduling (BS) [20] is a balanced-energy scheduling model designed for dense sensor networks. It distributes the sensing and communication tasks to all sensor nodes in the cluster. The assumptions made by the reviewed studies are listed in Table 1.

Table 1. The studies' classifications according to their assumptions.

	Assumptions								
Strategies	**Network Structure**	**Sensor Placement**	**High Density**	**Sensing Area**	**Time Synch**	**Frequent Failures**	**Mobility**	**Known Location**	**Known Distance**
RIS [18]	Flat	Grid, Uniform, Poisson	N	2-D	Y	N	N	N	N
BS [20]	Hierarchical	Poisson	N	2-D	Y	N	N	N	Y
LDAS [15]	Flat	Uniform	N	2-D	N	N	N	N	N
PEAS [16]	Flat	Uniform	Y	Any	N	Y	N	N	N
PECAS [17]	Flat	Uniform	N	Any	N	Y	N	N	N
CCP [21]	Flat	Any	N	2-D	N	N	N	Y	N
ASCENT [18]	Flat	Any	Y	Any	N	N	N	N	N
CSS [22]	Flat	Any	Y	2-D	Y	N	N	Y	N
LEACH-GA [23]	Hierarchical	Any	N	Any	Y	N	N	N	N
IBLEACH [24]	Hierarchical	Any	N	Any	Y	N	N	N	N
ESS [25]	Hierarchical	Any	N	Any	Y	N	N	N	N
DSS [26]	Hierarchical	Poisson	N	2-D	Y	N	N	N	Y
IDCA [27]	Flat	Any	N	2-D	N	N	N	Y	N

Y: yes; N: no.

Applications differ in their necessities. Hence, the served sensor networks have diverse design objectives and priorities. We summarize these design objectives as follows:

- Maximizing Network Lifetime. This is a goal that is hard not to consider.
- Sensing Coverage. A network is said to achieve k-coverage if any event occurs within the jurisdiction of at least k sensors. 1-overage is a minimum requirement for WSN.
- Network Connectivity. The developers applaud a model that offers a particular network connectivity needed by the application, but this requires a very high sensor density.
- Balanced Energy Usage. In the case of a sensing coverage breach, when a node runs out of power before others, some studies endeavor to spread the energy utilization evenly among each node.

- Simplicity. Sensors have exceptionally restricted memory space and limited computation power. For this reason, simple schemes are more popular.
- Robustness: Robustness measures how well a network can withstand downtime and crashes.

Adaptive self-configuring sEnsor networks topologies (ASCENT) [18], PEAS, PECAS, Low-energy adaptive clustering hierarchy (LEACH-GA) [23], IBLEACH [24], energy-efficient surveillance system (ESS) [25] and BS [20] do not take complete coverage of the region as their primary goal. Cooperative spectrum sensing (CSS) [22], however, is the opposite. The coverage configuration protocol (CCP) [21] introduced the concept of interconnected sensor coverage. It devised an approximation algorithm to construct a topology with near-optimal sensor coverage. The central controller periodically selects sensors along a trajectory until the target area is fully covered.

Most schemes strive to attain an energy balance [28], for example, the distributed self-spreading algorithm (DSS) [26] and intelligent deployment and clustering algorithm (IDCA) [27]. In the DSS, the sensor nodes are initially deployed randomly and move because of the influence exerted by nearby nodes. In the IDCA, however, the remaining energy level of the node determines whether it moves or not. The idea behind DSS and IDCA is to reduce the residual capacity differential between nodes.

Many studies envisage network connectivity with sensing coverage, for example, [29–31]. When the transmission range of the sensor node is at least twice its sensing range, k-coverage leads to k-connectivity [29,30]. Typically, high connectivity ensures high robustness, but one result of high connectivity is that data conflicts between nodes can seriously affect data transfer rates. The results of recent research presented in [32] are not based on the assumption that the transmit range of the sensor nodes is not less than twice their sensing range. Dhumal et al. [31] considered a tiered sensor network that includes sensors that could fail and they discussed the sensing coverage, network connectivity, and network diameter. In [19], the authors proposed an optimal deployment strategy to achieve two connections that fully cover all communication and sensing ranges.

The design objectives of the reviewed studies are summarized in Table 2.

Table 2. The studies' classifications according to their objectives.

Strategies	**Objectives**				
	Sensing Coverage	**Network Connectivity**	**Simplicity**	**Robustness**	**Energy Balance**
RIS [18]	k-coverage,asymptotic	×	×	Y	Y
BS [20]	×	×	×	×	Y
LDAS [15]	1-coverage	×	×	×	Y
PEAS [16]	×	1	×	×	×
PECAS [17]	×	1	×	×	Y
CCP [21]	k-coverage	k	×	×	Y
ASCENT [18]	×	×	Y	×	×
CSS [22]	k-coverage	k	×	Y	Y
LEACH-GA [23]	×	×	×	Y	Y
IBLEACH [24]	×	k	×	Y	Y
ESS [25]	×	×	×	Y	Y
DSS [26]	×	×	×	×	Y
IDCA [27]	Full-coverage	×	×	×	Y

×: not mentioned.

2.2. *Optimization Algorithms for Multi-Modal Function*

Population or single solution search-based optimization algorithms (viz., meta, hyper-heuristic) have solid global search ability. Typical examples include inter alia evolutionary and swarm intelligence algorithms, such as the genetic algorithm (GA) [33], differential evolution (DE) algorithm [34], particle swarm optimization (PSO) algorithm [35], comprehensive learning particle swarm optimization (CLPSO) [36] and artificial bee colony (ABC) algorithm [37]. These algorithms are essentially methods by trial and error. It can take many thousands or even millions of iterations to converge. New emerging algorithms, such as the multi-layered gravitational search algorithm (MLGSA) [38] and quantum tabu search (QTS) algorithm [39], improve classical versions by avoiding premature convergence or being trapped in local optima. When addressing practical issues, we search for the global extremum of a complicated or unknown function, but just finding one local minimum of a relatively simple but very high-dimensional function can also be a formidable challenge; for example, the Multi-modal Function Optimization (MFO) [40]. Given a multi-modal problem, the optimization task is to identify the greatest number of optimal solutions (global and local) to help the decision-maker better understand the problem at hand.

It has developed many techniques to locate local optimums. These techniques are termed "niching" methods. A niching method can be built into a standard search-based optimization algorithm to identify several optimal or sub-optimal solutions sequentially or concurrently. Sequential approaches discover optimal solutions gradually over time, while concurrent methods encourage and maintain many stable sub-populations within a single population. Conventional niching techniques include crowding, fitness sharing, decommissioning, covariance matrix adaptation, cleaning, species conserving, and so forth. More recently, some variants of meta-heuristic algorithms, such as an ant colony system with nonlinear pheromone update (ACS-NP) [41], and comprehensive learning particle swarm optimization with local search (CLPSO-LS) [42], have incorporated niching methods. Despite niching methods' first appearance over 30 years ago (in the 1980s), niching methods (or multi-modal optimization) are being resurrected as an increasingly important research subject, attracting researchers from a broad spectrum of research areas, including Evolutionary Computation (EC) [43] and Swarm Intelligence (SI) [42].

The Fibonacci tree optimization algorithm (FTO) [44] is a sophisticated optimization algorithm. It resolves the optimal solution of the problem by alternating iterations of global scanning and local scanning. It fully utilizes computer memory to save the optimization process. FTO can provide a reasonable approximation of a global optimum for a function with ample search space. In each iteration, the golden ratio separation is used to compress the search space, so the local optimal solution can also be achieved. It particularly applies to multi-peak/multi-modal function optimization.

The comparison of features of the aforementioned optimization algorithms is shown in Table 3.

Table 3. Comparison of some optimization algorithms.

Algorithm	Type	Searching Philosophy	Global Search Ability	Local Search Ability	Incorporate Niching Techniques	Convergence Speed
GA [33]	evolutionary intelligence	probabilistic search	◑	◑	No	Slow
DE [34]	evolutionary intelligence	probabilistic search	◑	○	No	Slow
PSO [35]	swarm intelligence	probabilistic search	◑	○	No	Slow
CLPSO [36]	swarm intelligence	probabilistic search	◑	○	No	Slow
ABC [37]	swarm intelligence	probabilistic search	◑	○	No	Slow
MLGSA [38]	swarm intelligence	probabilistic search	●	◑	No	Fast
ACS-NP [41]	swarm intelligence	probabilistic search	●	◑	Yes	Fast
CLPSO-LS [42]	swarm intelligence	probabilistic search	◑	◑	Yes	Fast
QTS [39]	evolutionary intelligence	probabilistic search	◑	●	No	Fast
FTO [44]	Recursive search	recursive search	●	●	No	Fast

○: weak; ◑: medium; ●: strong.

3. System Model

Table 4 lists the essential notations used in the system model.

Table 4. List of essential notations in the model descriptions.

#	Description
W	Total working duration
S	Total sleep duration
W_k	The k-th working interval
S_k	The k-th sleep interval
n	Number of intervals
S_0	Initial sleep interval (fixed)
s	Average sleep duration per duty cycle
w	Average working duration per duty cycle
M	The interested 2-D square region
M_x	Length of M
M_y	Width of M
$C(t)$	The counting process of events in $[0,t]$
$\hat{q}\vert_{[0,t]}$	Expected number of events processed in $[0,t]$
Γ_s	Average number of missed events in sleep duration
Γ_w	Average number of missed events in working duration
Γ	Event space
λ	Poisson rate
d	Distance between sensor and event
r	Sensing radius
r_{min}	Minimum sensing radius
r_{max}	Maximum sensing radius
μ	Detection accuracy
ξ_1, ξ_2	Metrics of detection ability
e	Energy depletion for sensing
e_S	Energy depletion of sensing in sleep duration
e_W	Energy depletion of sensing in working duration
ε_S	Static power loss in sleep mode
ε_W	Static power loss in working mode
γ	Coefficient of energy depletion growing with distance
α	Changeable power scaling parameter for the sensing circuit
δ	Step-size increasing factor
θ	Step-size diminishing factor
N	Number of sensors
$Sgn(\cdot)$	The sign function
$\sigma(\cdot)$	The logistic Sigmoid function

3.1. Basic Assumptions

The model operates on a 2-D map with many devices, each equipped with sensors to perform specific tasks. We make the following basic assumptions:

- **A1**: The network structure is flat. All nodes are homogeneous with the same energy budget.
- **A2**: The positions of sensor nodes are uniformly distributed.
- **A3**: The sensing area is 2-D.
- **A4**: The transmission/sensing radii are tunable.
- **A5**: Time is asynchronous. It is hard to coordinate sensors without a central controller but, otherwise, the central controller incurs a performance penalty. So, sensors embrace asynchronous scheduling—every sensor autonomously decides its duty cycle without synchronizing with each other.
- **A6**: The sensor node can be movable.
- **A7**: Location information is unknowable.
- **A8**: Distance information is unknowable.

3.2. Location Distribution and Sensing Radii

We assume the participants move around in M randomly. The locations of all N participants follow a uniform distribution. To accommodate different sensing radii, the sensor can adjust its transmission range r from r_{min} to r_{max}, which is usually accomplished by varying the transmitting power (see Figure 1).

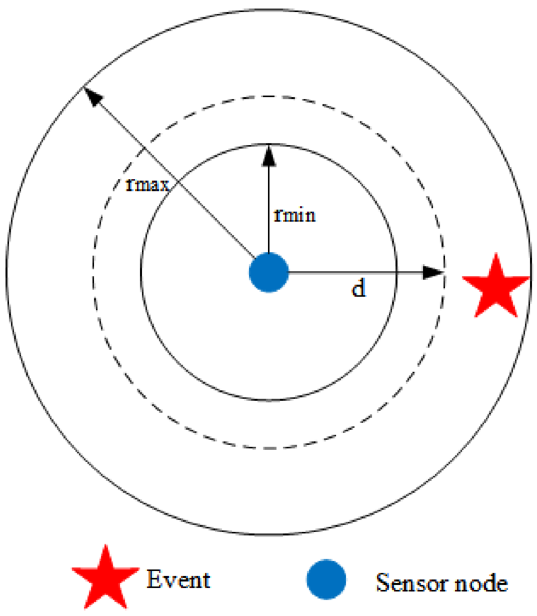


Figure 1. Tunable sensing radii.

3.3. Event Occurrence and Detection

We assume the events occur randomly and independently in M, subject to a Poisson process of Poisson rate λ. The Poisson process is a stochastic process used to model random and independent events. Furthermore, it is assumed that the measurement is efficient, which means all the events occurring within the sensing range will be expediently detected. Therefore, the probability that q events processed by a sensor with sensing radius r in duration t is

$$\Pr\left(C(t)=q\right)=\frac{\exp\{-\lambda'\pi r^2 t\}\left(\lambda'\pi r^2 t\right)^q}{q!}=\frac{\exp\{-\lambda t\}(\lambda t)^q}{q!}, \tag{1}$$

where $C(t)$ is the counting process of events and $\lambda=\lambda'\pi r^2$ represents the average Poisson rate in the circular sensing region of the sensor. By leveraging the properties of Poisson's process, we have:

$$\mathbb{E}[C(t)]=\hat{q}|_{[0,t]}=\lambda t, \tag{2}$$

where $\mathbb{E}[\cdot]$ is the expectation operator and $\hat{q}|_{[0,t]}$ denotes the expected number of events processed in $[0,t]$.

An event at a distance of d from the sensor has a probability of being detected:

$$\Pr(d)=\begin{cases}1, & \text{if } d<r_{min}\\ \exp\left\{-\xi_1(d-r_{min})^{\xi_2}\right\}, & \text{if } r_{min}<d<r_{max}\,,\\ 0, & \text{if } d>r_{max}\end{cases} \tag{3}$$

where ξ_1 and ξ_2 are parameters determined by the sensor physical characteristics, which reflects the probability attenuation coefficients with distance.

In sleep time, all events are considered missed. Let Γ_S be the average number of missed events, then

$$\Gamma_S=\mathbb{E}[C(S)]=\lambda S. \tag{4}$$

During working hours, the average number of missed events, Γ_W, can be calculated as

$$\Gamma_W = \mathbb{E}[C(W)](1 - \Pr(d)) = \lambda W(1 - \Pr(d)). \tag{5}$$

The event space is

$$\Gamma = \mathbb{E}[C(W + S)] = \lambda(W + S). \tag{6}$$

Now we define the detection accuracy μ as a quantifiable indicator of event missing

$$\mu = 1 - \frac{\Gamma_S + \Gamma_W}{\Gamma}. \tag{7}$$

It follows from (3)–(6) that

$$\mu = \begin{cases} \dfrac{W}{S+W}, & \text{if } d < r_{min} \\ \dfrac{W}{S+W} \exp\left\{-\xi_1 (d - r_{min})^{\xi_2}\right\}, & \text{if } r_{min} < d < r_{max} \\ 0, & \text{if } d > r_{max} \end{cases}. \tag{8}$$

3.4. Energy Depletion

We consider the sensing energy model introduced in Lee et al. [8]:

$$e_W = \gamma(d - r_{min})^{\alpha} + \varepsilon_W, \tag{9}$$

where γ, α are sensor-specific constants; in exemplary cases, $\alpha \in [2, 4]$; ε_W describes the static power and the wake-up timing and wake-up frequency will affect the static power consumption, so ε_W is inextricably related to scheduling strategies. Equation (9) implies that the energy depletion of the perceptron during the working duration is determined by the distance at which an object appears. If a sensor is in sleep mode, then only the low-power timer and other necessary components remain active, viz.

$$e_S = \varepsilon_S; \tag{10}$$

ε_S represents the static power. Since frequent wake-up and sleep can affect the static power, that means ε_S is also a function of the scheduling strategy.

The total energy depletion for the sensing circuit is going to be the weighted sum of these two,

$$e = e_S S + e_W W, \tag{11}$$

where W and S represent working duration and sleeping duration, respectively. Then the mathematical expectation is:

$$\mathbb{E}[e] = e_S \frac{S}{S+W} + e_W \frac{W}{S+W}. \tag{12}$$

Assign (9) and (10) to (11), we have:

$$\mathbb{E}[e] = \varepsilon_S \frac{S}{S+W} + \left(\gamma(d - r_{min})^{\alpha} + \varepsilon_W\right) \frac{W}{S+W}. \tag{13}$$

3.5. Scheduling Model

Figure 2 shows a schematic diagram of a typical alternate sensing scheduling scheme. A sensor wakes up its sensing unit when transitioning to the working state and turns it off when the sleep interval comes, and the length of the sleep interval is dynamically adjusted according to the event occurrence of the previous interval. Despite the scheme's prevalence, how to adjust the intervals properly is as yet an open issue.

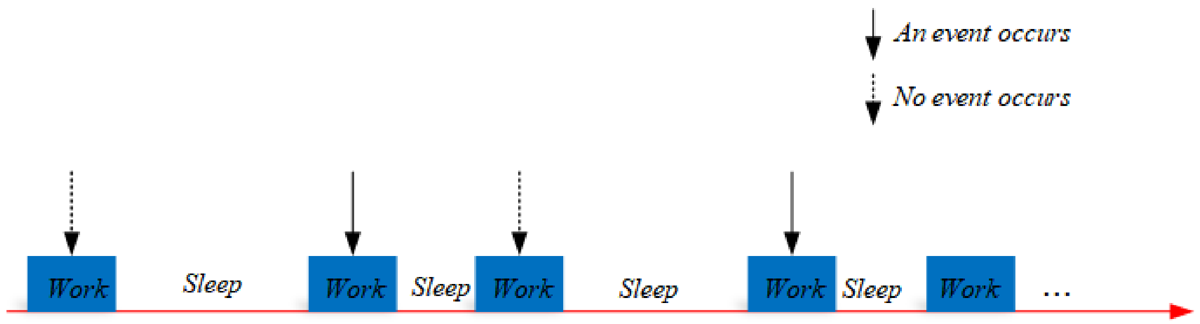


Figure 2. The diagram of alternate sensing scheduling.

Now, we establish a baseline scheduling strategy: fixed-time scheduling (denoted as FT for convenience). In FT, we set the working duration and the sleep duration in advance and they remain unchanged. Then, we introduce an adaptive scheduling strategy, in which we dynamically fine-tune the sleep duration as per the event statistics and user-specified parameters. The mathematical formula is:

$$S_k = \begin{cases} S_{k-1} + \delta S_{k-1}, & Sgn(W_{k-1}) = 0 \\ S_{k-1} - \theta S_{k-1}, & Sgn(W_{k-1}) = 1 \end{cases}, \tag{14}$$

where δ and θ denotes the step-size increasing/diminishing factor, $0 \le \delta \le 1$, $0 \le \beta \le 1$. $Sgn(\cdot)$ is sign function:

$$Sgn(t) = \begin{cases} 1, & \text{If an event is detected during time } t \\ 0, & \text{Otherwise} \end{cases}. \tag{15}$$

δ and θ provide the ability to maintain a synergy between the event capture and energy expenditure. For instance, a larger δ renders more energy-saving but also increases the probability of missing an event.

3.6. Sensing Coverage

Practical applications require the monitoring of almost the whole area of the wireless sensor network. Achieving 1-coverage is a minimum requirement (as shown in Figure 3).

Theorem 1. *Suppose there are N sensors uniformly distributed in a WSN, the sensing radius of each sensor is r, $r \in [r_{min}, r_{max}]$, then the overall detection accuracy is given by:*

$$\tilde{\mu} = 1 - \exp\left\{-\frac{W}{S+W}\frac{N\pi r^2}{M_X M_Y}\right\}. \tag{16}$$

Proof. The probability of a random event occurring within the sensing radii of N_0 sensors is:

$$\Pr{}_0 = \frac{\left(\frac{N\pi r^2}{M_X M_Y}\right)^{N_0}}{N_0} \exp\left\{-\frac{N\pi r^2}{M_X M_Y}\right\}. \tag{17}$$

The expected number of sensors within the scope of an event is:

$$\mathbb{E}[N_0] = \frac{N}{M_X M_Y}\pi r^2. \tag{18}$$

The expected number of sensors that are within the scope of an event and are themselves in a working state is:

$$\mathbb{E}[N_0^W] = -\frac{W}{S+W}\frac{N}{M_X M_Y}\pi r^2. \tag{19}$$

Therefore, the probability of a random event being detected by a sensor is:

$$1 - \exp\left\{ -\frac{W}{S+W} \frac{N\pi r^2}{M_X M_Y} \right\}. \tag{20}$$

The theorem holds. $\square$

Theorem 1 reveals that the detection accuracy is a function of the average number of sensors in each cell area and sensing scheduling.

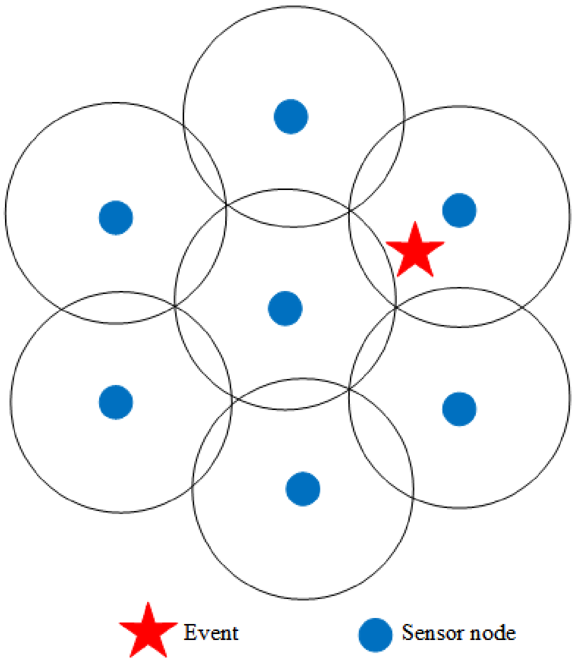


Figure 3. The diagram of 1-coverage.

3.7. Problem Formulation

The design objective of our model is an optimization problem:

$$\min J = \min\left\{ \frac{1}{2}(1-\mu) + \frac{1}{2}\sigma\left(\frac{\mathbb{E}[e]}{1000} \right) \right\}. \tag{21}$$

We change the optimization goal from maximizing μ to minimizing $1-\mu$. $\sigma(\cdot)$ is the logistic Sigmoid function, $\frac{1}{1+\exp(-x)}$, which maps the entire number line into a small range between 0 and 1. The reason we divide $\mathbb{E}[e]$ by 1000 is that we want to convert the value of $\mathbb{E}[e]$ to the linear region of the Sigmoid function—as can be seen from Figure 4, the linear region of $\frac{1}{1+\exp(-x)}$ is only 0–1 mW, while that of $\frac{1}{1+\exp(-x/1000)}$ is 0–4000 mW wide, which can fully cover the maximum variation range of energy depletion.

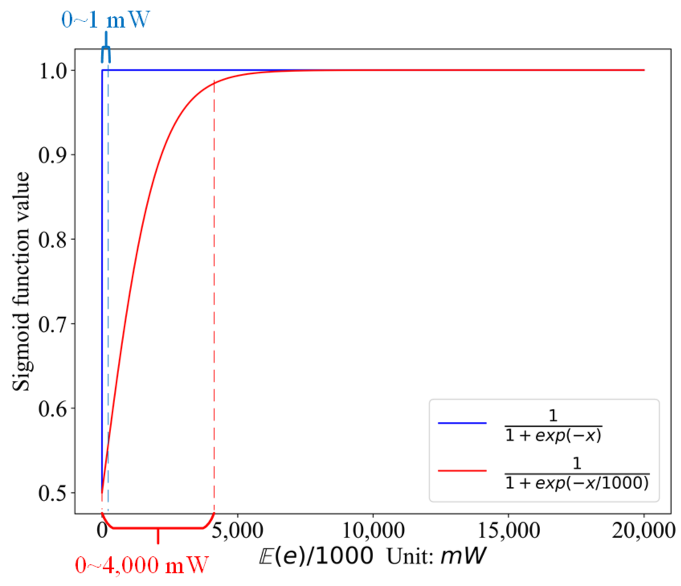


Figure 4. The liner region for different Sigmoid functions.

We list all the equations that determine J:

$$\begin{cases} S = \sum_{n}^{k=1} S_k \\ S_k = \begin{cases} S_{k-1} + \delta S_{k-1}, & Sgn(W_{k-1}) = 0 \\ S_{k-1} - \theta S_{k-1}, & Sgn(W_{k-1}) = 1 \end{cases} \\ W = \sum_{n}^{k=1} W_k \\ W_k = \bar{w} \\ \mu = \begin{cases} \frac{W}{S+W}, & \text{if } d < r_{min} \\ \frac{W}{S+W} \exp\left\{-\xi_1 (d - r_{min})^{\xi_2}\right\}, & \text{if } r_{min} < d < r_{max} \\ 0, & \text{if } d > r_{max} \end{cases} \\ \mathbb{E}[e] = \varepsilon_S \frac{S}{S+W} + \left(\gamma (d - r_{min})^{\alpha} + \varepsilon_W\right) \frac{W}{S+W} \\ d \le M_X \\ d \le M_Y \\ 2 \le \alpha \le 4 \\ 0 \le \delta \le 1 \\ 0 \le \theta \le 1 \end{cases} \tag{22}$$

Since the occurrence of events is random, then:

$$\Pr\{Sgn(W_{k-1}) = 0\} = \Pr\{Sgn(W_{k-1}) = 1\} = \frac{1}{2}, \tag{23}$$

which means

$$\begin{aligned} S &= \sum_{k=1}^{n} S_k \\ &= \sum S_k | \{Sgn(W_{k-1}) = 0\} + \sum S_k | \{Sgn(W_{k-1}) = 1\} \\ &= \left(1 + \frac{\delta - \theta}{2}\right) n S_0. \end{aligned} \tag{24}$$

The working interval is fixed to ϖ, so

$$W = \sum_{k=1}^{n} W_k = n\varpi. \tag{25}$$

Put these together, then we get the final objective function:

$$J = \begin{cases} \frac{1}{2}\frac{\left(1+\frac{\delta-\theta}{2}\right)nS_0}{\left(1+\frac{\delta-\theta}{2}\right)nS_0+n\varpi} + \frac{1}{2}\sigma\left(\frac{1}{1000}\frac{(\gamma(d-r_{min})^{\alpha}+\varepsilon_W)n\varpi+\varepsilon_S\left(1+\frac{\delta-\theta}{2}\right)nS_0}{\left(1+\frac{\delta-\theta}{2}\right)nS_0+n\varpi}\right), & \text{if } d < r_{min} \\ \frac{1}{2}\left(1-\frac{n\varpi\exp\left\{-\xi_1(d-r_{min})^{\xi_2}\right\}}{\left(1+\frac{\delta-\theta}{2}\right)nS_0+n\varpi}\right) + \frac{1}{2}\sigma\left(\frac{1}{1000}\frac{(\gamma(d-r_{min})^{\alpha}+\varepsilon_W)n\varpi+\varepsilon_S\left(1+\frac{\delta-\theta}{2}\right)nS_0}{\left(1+\frac{\delta-\theta}{2}\right)nS_0+n\varpi}\right), & \text{if } r_{min} < d < r_{max}\,. \\ \frac{1}{2}\sigma\left(\frac{1}{1000}\frac{(\gamma(d-r_{min})^{\alpha}+\varepsilon_W)n\varpi+\varepsilon_S\left(1+\frac{\delta-\theta}{2}\right)nS_0}{\left(1+\frac{\delta-\theta}{2}\right)nS_0+n\varpi}\right) & \text{if } d > r_{max} \end{cases} \tag{26}$$

The constraint conditions are

$$\text{s.t.} \quad \begin{array}{l} 0 \le \delta \le 1, \\ 0 \le \theta \le 1 \end{array}. \tag{27}$$

ε_W and ε_S are affected by the scheduling strategy. The question now is how to establish the functional relationship between ε_W, ε_S and δ, θ. We have been thinking about numerical fitting methods—if we have real historical sample sensory readings, we can fit out the function between ε_W, ε_S and δ, θ.

Numerical fitting is also known as curve fitting. As for the discrete sensory data collected by sampling, we often want to get a continuous function (i.e., a curve) or a denser discrete equation consistent with the known data. The steps of numerical fitting are:

1. Presume a functional form. Commonly used function forms include: polynomial function, exponential function, logarithmic function, trigonometric function, and so forth.
2. Determine the indeterminate coefficients. Using the least square method, point group center method, random fuzzy method, and so on, to determine the coefficients of the function. Of those, the least square method is the most commonly used.
3. Evaluation. The imitative effect can be measured by the mean square error (MSE) or the degree of fit (R2).

4. Fibonacci Tree Optimization Algorithm

Recall that one-dimensional Fibonacci (golden ratio) search cannot proficiently take care of multi-variate issues, giving rise to Fibonacci Tree Optimization—an improved multi-dimensional search algorithm [44].

Let $\mathbf{X_A}$, $\mathbf{X_B}$ and $\mathbf{X_C}$ be the vectors in D-dimensional Euclidean space. In particular, $D = 2$. $\mathbf{X_A}$ and $\mathbf{X_B}$ address the endpoints of the search component fulfilling the optimization rule, and $\mathbf{X_C}$ signifies the split points that can be resolved from searching rules. A proportion of the vectors can be determined as follows:

$$\frac{\|\mathbf{X_C}-\mathbf{X_A}\|}{\|\mathbf{X_B}-\mathbf{X_A}\|} = \frac{\|\mathbf{X_B}-\mathbf{X_C}\|}{\|\mathbf{X_C}-\mathbf{X_A}\|} = \frac{F_p}{F_{p+1}}, \tag{28}$$

where F_p denotes the pth Fibonacci number, and $\frac{F_p}{F_{p+1}}$ is equal to what is called the golden ratio.

The fitness function determined by the endpoints in the construction ought to be assessed as:

$$J(\mathbf{X_A}) < J(\mathbf{X_B}). \tag{29}$$

$J(\cdot)$ denotes the objective function. Then, the coordinate for split point $\mathbf{X_C}$ can be calculated as:

$$\mathbf{X_C} = \mathbf{X_A} + \frac{F_p}{F_{p+1}}(\mathbf{X_B} - \mathbf{X_A}). \tag{30}$$

Finding the optimal value can likewise be viewed as setting up a search component in FTO, and can be partitioned into two phases: the local optimization phase and the global optimization phase. Allow Θ to signify the points set of the objective function and set $|\Theta|_{num} = F_p, i = 1, 2, \cdots, n$, where $|\cdot|_{num}$ denotes the number of points in the set, and d represents the depth of the tree. We select the point with the best fitness value. Then, in the following optimization stage, the points are rearranged by their fitness values from best to worst.

There are two search rules in FTO, which can be summed up:

Rule One: Consider the endpoints $\mathbf{X_A}$ and $\mathbf{X_B}$, which are given by

$$\{\mathbf{X_A}\} = \Theta = \{\mathbf{X_k} | k \in [1, F_p]\} \tag{31}$$

$$\{\mathbf{X_B}\} = \Theta = \left\{\mathbf{X} | \mathbf{X} \in \prod_{j=1}^{D}\left[X_{lb}^J, X_{ub}^J\right]\right\}, \tag{32}$$

where Θ denotes the points set, with each point represented as $\mathbf{X_k}$, of the pth iteration, and k is the index. $\mathbf{X_A}$ absorb every point from Θ. $\mathbf{X_B}$ select F_p points from Θ randomly, where F_p denotes the population size. X_{ub}^J and X_{lb}^J are the upper and lower limits for every point. For $\forall \mathbf{X} \in \{\mathbf{X_B}\}$, $\mathbf{X}$ satisfies a uniform distribution over the span $\left[X_{lb}^J, X_{ub}^J\right]$, viz.

$$\Pr(\mathbf{X}) = U(X_{lb}, X_{ub}) = \frac{1}{X_{ub} - X_{lb}}. \tag{33}$$

Using the $\mathbf{X_A}$ and $\mathbf{X_B}$ given above, the split points $\mathbf{X_{S1}}$ are solved by Equation (30).

Rule Two: Assume that $\mathbf{X_{best}}$ is the optimal point in the current iteration given by rule one, viz.

$$\mathbf{X_{best}} = BEST(\Theta). \tag{34}$$

Then, let $\mathbf{X_A} = \mathbf{X_{best}}$; we have

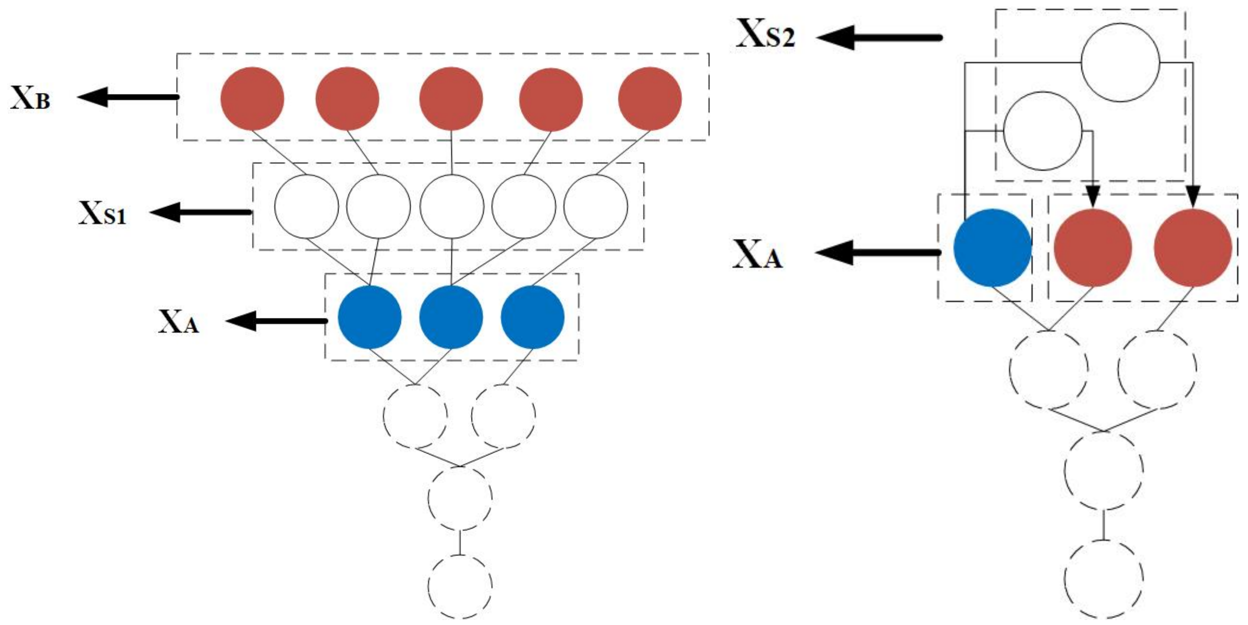
$$J\{\mathbf{X_A}\} = \min\{J\{\mathbf{X_k}\} | k \in [1, F_p]\} \tag{35}$$

$$\mathbf{X_B} = \{\mathbf{X_k} | \mathbf{X_k} \in \Theta \wedge \mathbf{X_k} \neq \mathbf{X_A}\}. \tag{36}$$

Hence, the split points $\mathbf{X_{S2}}$ in the local optimization stage can be determined by Equations (35) and (36).

After applying the two rules above, new endpoints $\mathbf{X_A}$ and $\mathbf{X_B}$ and split points $\mathbf{X_{S1}}$ and $\mathbf{X_{S2}}$ are generated, and now we have $3F_p$ points. These points are sorted from best to worst based on the fitness value, retaining the optimal $F_p + 1$ points, while the remaining $3F_p - F_p - 1$ points are eliminated. At the end of this process, the set of search spaces for the current p iteration is updated from the remaining points and form a new set for the next iteration.

Figure 5 shows the branch generation process in the Fibonacci tree optimization algorithm. The depth is initialized as expected, and the number of points in each branch layer is equal to F_p. The dotted circle represents the search points set of the previous iteration; the solid red circle represents the endpoint $\mathbf{X_A}$ of the current iteration, and the solid blue circles represent the global random endpoints. Figure 5a shows the global optimization phase, the split points $\mathbf{X_{S1}}$ are constructed based on $\mathbf{X_B}$ and $\mathbf{X_A}$ and are represented by a solid white circle. Figure 5b shows the local optimization phase. New split points $\mathbf{X_{S2}}$ can be obtained as per the rules.



(a) global optimization stage (b) local optimization stage

Figure 5. The diagram of the branch generation process in the Fibonacci tree optimization algorithm.

The pseudo-code for the Fibonacci branch generation process is shown as Algorithm 1.

Algorithm 1: Pseudo-code of Fibonacci branch generation process

```
Input: Branch depth: $d$; Max iterations: $I_{max}$
Output: Optimal point: $\mathbf{X_{best}}$
Initialize set $\Theta$ with $F_1$ global random points;
for $p = 1 \to I_{max}$ do
    Rule 1:
    $\{\mathbf{X_A}\} \leftarrow \Theta$;
    Generates $\{\mathbf{X_B}\}$, $|\{\mathbf{X_B}\}|_{num} = F_p$;
    Compute the split points set $\{\mathbf{X_{S1}}\}$ and merge them into the set $\Theta$;
    Sort and preserve top best $F_p$ points of the set $\Theta$;
    Rule 2:
    $\mathbf{X_A} \leftarrow \mathbf{X_{best}}$;
    $\{\mathbf{X_B}\} = \{\mathbf{X}_k | \forall \mathbf{X}_k \in \Theta \wedge \mathbf{X}_k \neq \mathbf{X_A}\}$;
    Compute the split points set $\{\mathbf{X_{S2}}\}$ and merge them into the set $\Theta$;
    Sort and preserve top best $F_p$ points of the set $\Theta$;
    $p \leftarrow p + 1$;
    if $|\Theta|_{num} = F_d$ then
        break;
    end
end
return $\mathbf{X_{best}}$;
```

5. Analysis

5.1. Why FTO Algorithm Is Chosen for Parameter Optimization?

1. In the FTO algorithm, the growth of a Fibonacci tree is based on the optimal points of the latest generation, which is a process of competitive elimination. If there are peaks with different heights, the smaller peaks will be removed in the optimization process. Hence, the global optima can be achieved.
2. The inner structure of the FTO algorithm meets the golden ratio all over. With each iteration, the separation of the golden ratio rapidly compresses the search space,

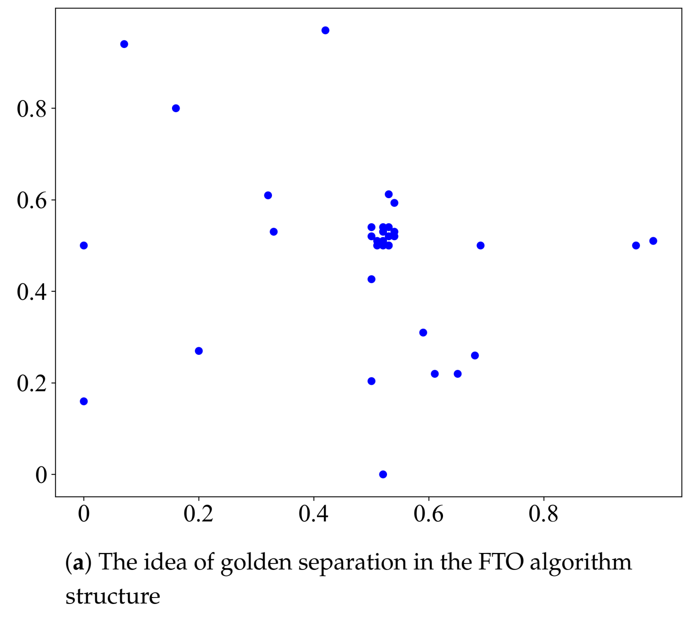
providing the local optimal solution. It is adapted to the optimization of multi-modal functions. Conversely, most heuristic algorithms are based on the probabilistic search; they are essentially trial and error methods. It may take thousands or even millions of iterations, which is uneconomical and slow.

3. The FTO setup only concerns the initial Fibonacci number, which is exquisite and simple. The typical way to obtain multiple local extremums of the multi-modal function using heuristic algorithms is the niche technology. Setting population parameters has a significant impact on the optimistic effects.
4. Just like most heuristic algorithms, FTO is also inspired by natural phenomena—plant phototropism. Plants flourish more in sunny areas. To solve the optimization of the multi-modal function, FTO considers the global search space as a sunlit area. The hit probability is sustained and stable. The lush areas refer to the local search process, reflecting the fact that FTO can alternate between global and local searches.
5. It fully utilizes the computer memory to save the optimization process and therefore is traceable.

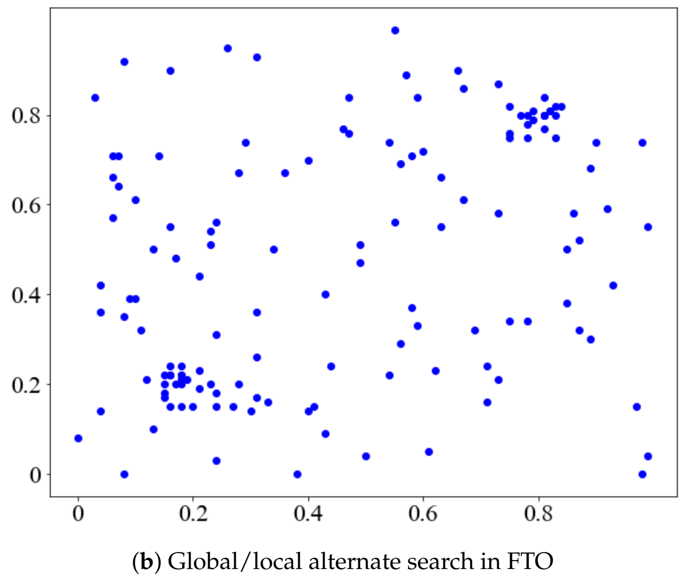
The comparison between FTO and other optimization algorithms is listed in Table 5. Figure 6a illustrates the idea of golden separation in the FTO algorithm's structure. Figure 6b reflects the global/local alternate search process inspired by phototropism of plants.

Table 5. The comparison between FTO and other optimization algorithms.

	FTO	**Heuristic Algorithms**
Parameters	simple	many, hard to determine
Imitation of nature	phototropism	swarm intelligence
Characteristic	competitive elimination alternate search golden ratio search	random search based on probability
Global search capability	strong	prone to be trapped in local optimum
Convergence speed	fast	slow
Resources footprint	low	high



(**a**) The idea of golden separation in the FTO algorithm structure



(**b**) Global/local alternate search in FTO

Figure 6. Optimal points in each iteration of FTO for optimizing: (**a**) $f(x,y) = \left(\frac{3}{0.05+9x^2+9y^2}\right)^2 + (9x^2+9y^2)^2$, and (**b**) $f(x,y) = 0.5(\sin(10\ln(x)) + \sin(10\ln(y))), x, y \in [0,1]$.

5.2. Why Do We Claim FTOS Strategy Is Superior to Other Scheduling Strategies?

1. Some of the latest scheduling strategies, such as LEACH, work under strict assumptions; for example, the network structure is flat, nodes are densely distributed, time is synchronized, and so forth. The FTOS strategy does not require time synchronization (centralized scheduling); each node independently determines the duty cycle; node density is not important; the location and distance information is not needed.
2. The FTOS strategy determines the scheduling parameters as per the occurrence of events from the previous period, which is very scalable.
3. Once the parameters are determined, they remain unchanged in the coming hours; you can easily deploy them.

Table 6 enlists the above-mentioned points.

Table 6. The differences between FTOS and other scheduling strategies.

		FTOS	**Other Scheduling Strategies**
Assumptions	Time synchronization	no	yes
	Node density	no requirement	high
	Transmission radii	tunable	untunable
	Node location	movable	immovable
Adaptability		strong	weak
Deployment		easy	difficult

6. Experiment

6.1. Experimental Setup

We set up a wireless sensor network comprising three nodes ($N = 3$), covering a 5 m $\times$ 5 m lawn area (see Figure 7). Each node owns a single-chip microcomputer with a STM32F103ZET6 processor and a diffuse reflection photoelectric infrared proximity sensor. The technical parameters of the sensor are presented in Table 7.

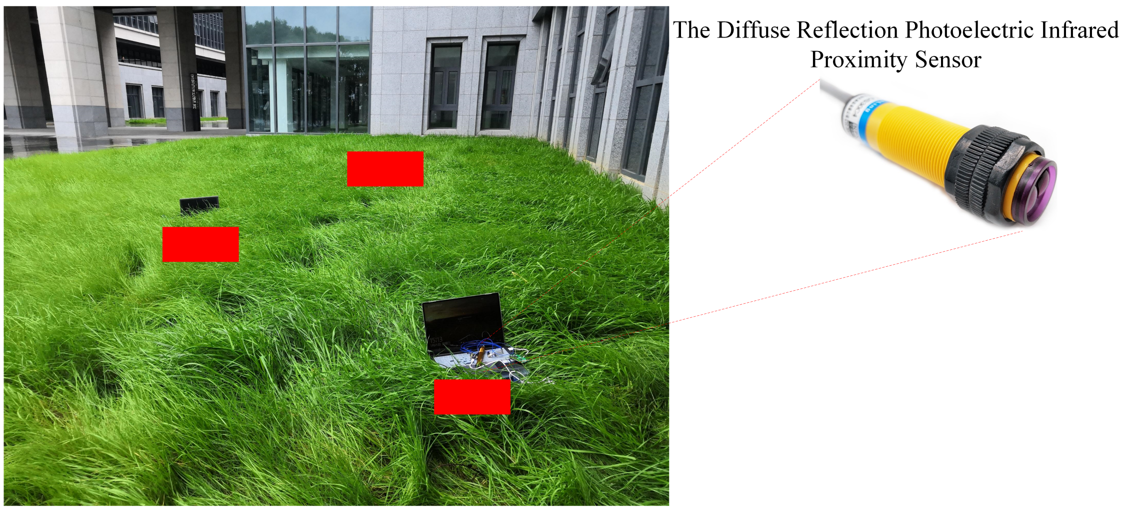

The Diffuse Reflection Photoelectric Infrared Proximity Sensor

Figure 7. The experimental scene.

We randomly throw objects into the area to simulate the event. The diffuse reflection photoelectric infrared proximity sensor with a sensing radius of r = 3~5 m ($r_{min} = 3, r_{max} = 5$) opens the switch when there is an object to block; the processor then issues a command to pull the buzzer.

Since the working voltage of the single-chip microcomputer is fixed to 5 V, the power formula is $P = UI$, so that the current reflects the size of the power consumption. Consequently, the energy is measured by the reading of the ammeter placed on each microcontroller.

Table 7. The technical parameters for the diffuse reflection photoelectric infrared proximity sensor.

Parameter	Value
Product model	E3F-DS100P1
Operating voltage	DC 6~36 V
Operating current	200 mA
Response frequency	50 Hz
Sensing object	Any opaque object
Sensing radius	10~100 cm, adjustable

6.2. Train

6.2.1. Function Fitting

We assume that the sensor node determines its sleep interval based on the adaptive scheduling model mentioned in Section 3, where the parameters δ and θ are randomly generated on $[0,1]$. The polynomial function, double-exponential function, and checkmark function are used to fit the data (the forms are shown in Table 8). The determination of the coefficients is performed by the nonlinear least square method.

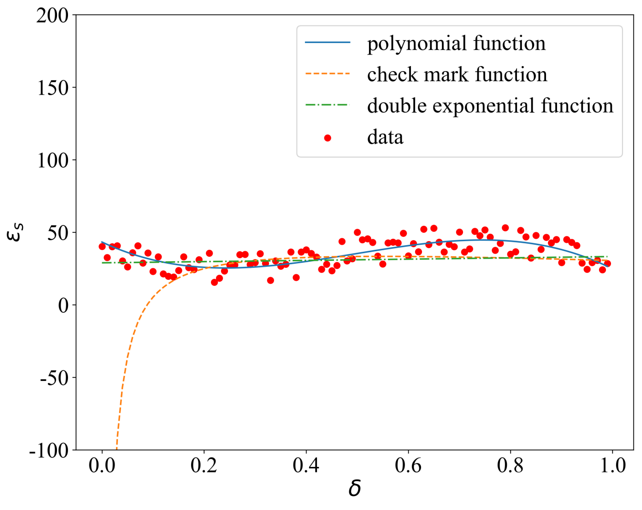
We generated 100 samples, recorded the static power losses, ε_S and ε_W, and the corresponding parameters δ and θ . Figure 8 shows the fitting curves. Table 9 enlists the degree of fit, R^2.

Table 8. The fitting functions.

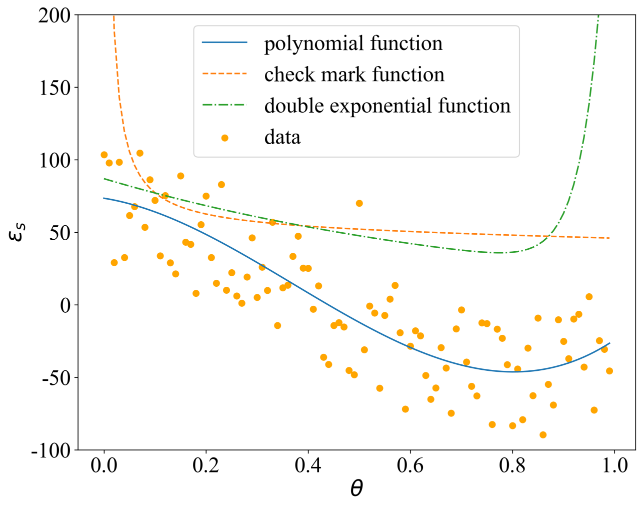
Function	Form
Polynomial function	$ax^5+bx^4+cx^3+dx^2+ex+f$
Double-exponential function	$ae^{bx}+ce^{dx}$
Checkmark function	$ax+\frac{b}{x}+c$

Table 9. The degree of fit (R^2).

Cases	Polynomial Function	Double-Exponential Function	Checkmark Function
ε_S and δ	0.87	0.93	0.35
ε_S and θ	0.83	0.80	0.56
ε_W and δ	0.73	0.32	0.46
ε_W and θ	0.54	0.76	0.93

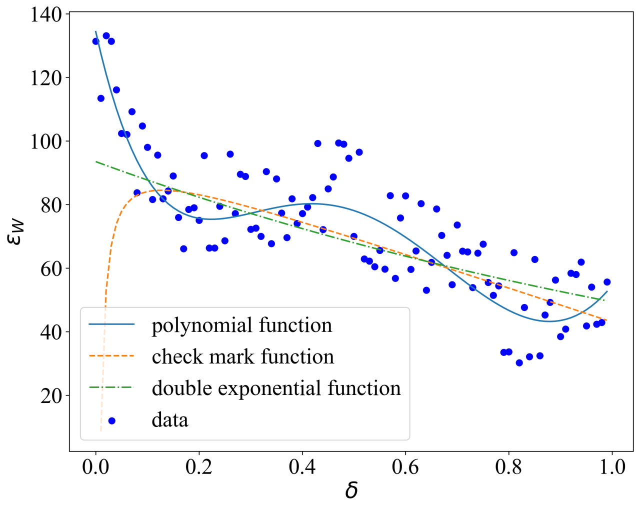


(**a**) ε_S and δ

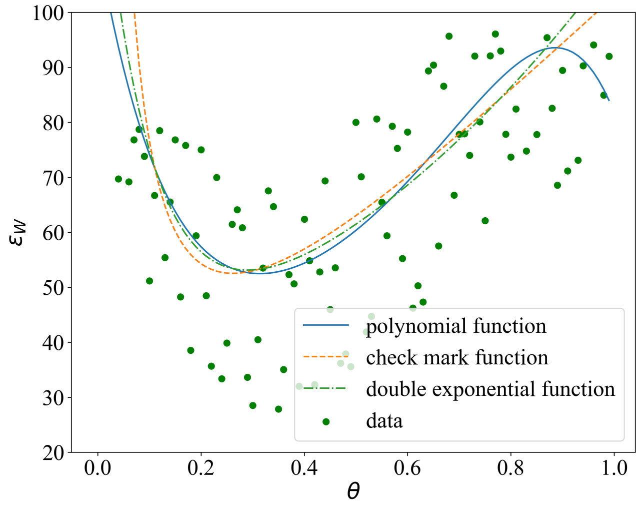


(**b**) ε_S and θ

Figure 8. *Cont.*



(**c**) ε_W and δ



(**d**) ε_W and θ

Figure 8. The fitting curves of ε_S, ε_W and δ, θ.

Now that we have determined the functional relations between ε_S, ε_W and δ, θ, the explicit form of the objective function J can be obtained by putting the fitted functions in.

6.2.2. Univariate Analysis

Three sets of parameters, (0.175, 0.175), (0.65, 0.65), (0.5, 0.5), were used to test the sensing power dissipation against α in base sleep duration $\bar{s}$ and base working duration $\bar{w}$. The result is shown in Figure 9. When the distance between the event and the sensor, and the base sleep duration and base working duration, are fixed ($d = 3.15$ m, $\bar{w} = 1$ s, $\bar{s} = 1$ s), the working power increases exponentially with α, and the sleeping power dissipation remains unchanged for α. With the increase of step-size parameter δ and θ, both the working power dissipation $e_{\bar{w}}$ and the sleeping power dissipation $e_{\bar{s}}$ decrease.

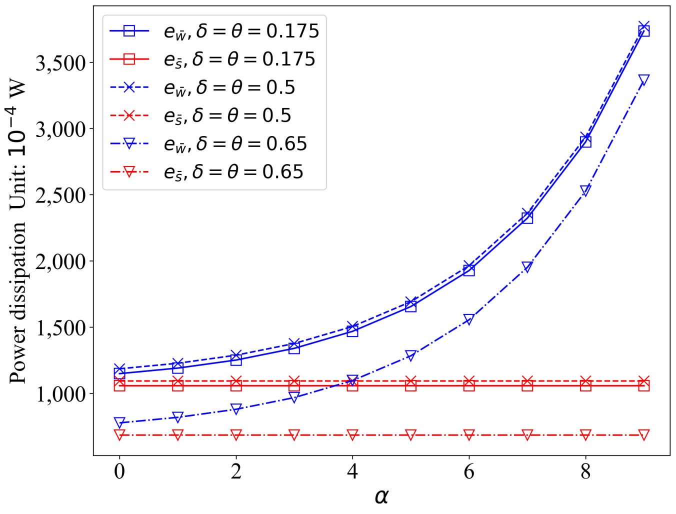


Figure 9. The power dissipation against α in base duration $\bar{s}$ and $\bar{w}$ ($d = 3.15$ m, $\bar{w} = 1$ s, $\bar{s} = 1$ s).

Figure 10 shows the variation of detection accuracy and expected energy depletion with δ. Figure 11 shows the variation of detection accuracy and expected energy depletion with θ. It can be seen that the two parts of the objective function J, $1 - \mu$ grow with δ, while $\sigma(\mathbb{E}[e]/1000)$ decreases with δ; the optima can only be obtained by jointly optimizing $1 - \mu$ and $\sigma(\mathbb{E}[e]/1000)$. $1 - \mu$ and $\sigma(\mathbb{E}[e]/1000)$ move in the opposite direction against

θ and $\bar{w}$ and $\bar{s}$ impose opposite effects on $1-\mu$ and $\sigma(\mathbb{E}[e]/1000)$. Anything that can increase/decrease $1-\mu/\sigma(\mathbb{E}[e]/1000)$ by increasing $\bar{s}$ can also be achieved by decreasing $\bar{w}$.

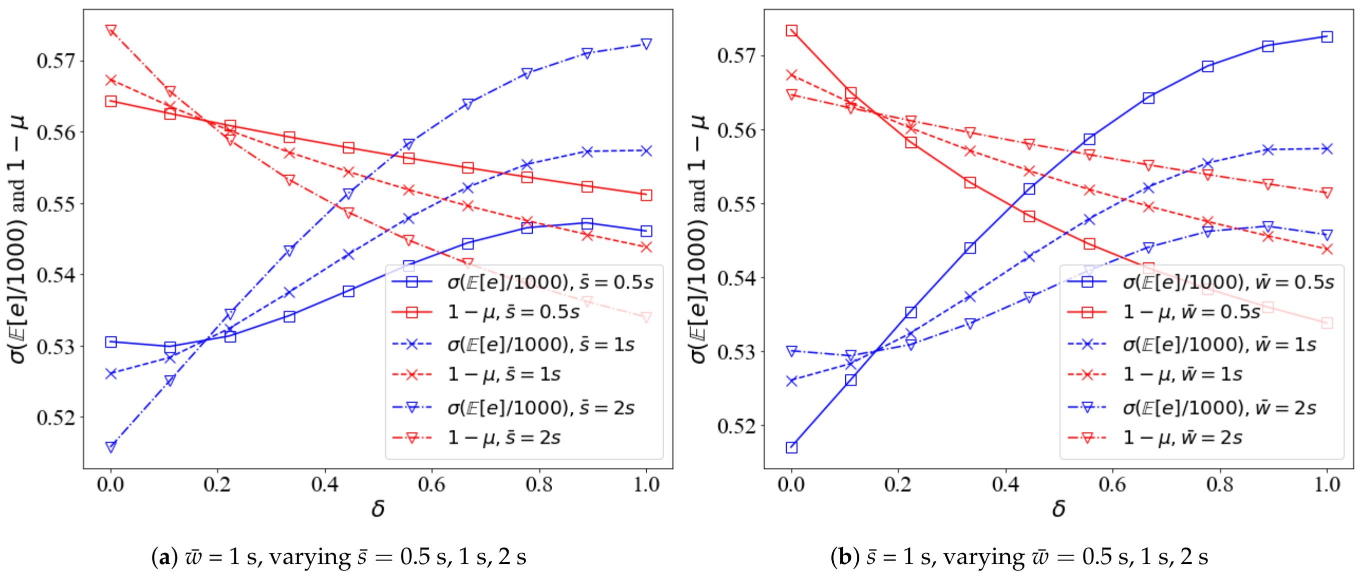


Figure 10. The detection accuracy and energy depletion against δ. (**a**) $\bar{w}$ is fixed to 1 s, varying $\bar{s}$; (**b**) $\bar{s}$ is fixed to 1 s, varying $\bar{w}$.

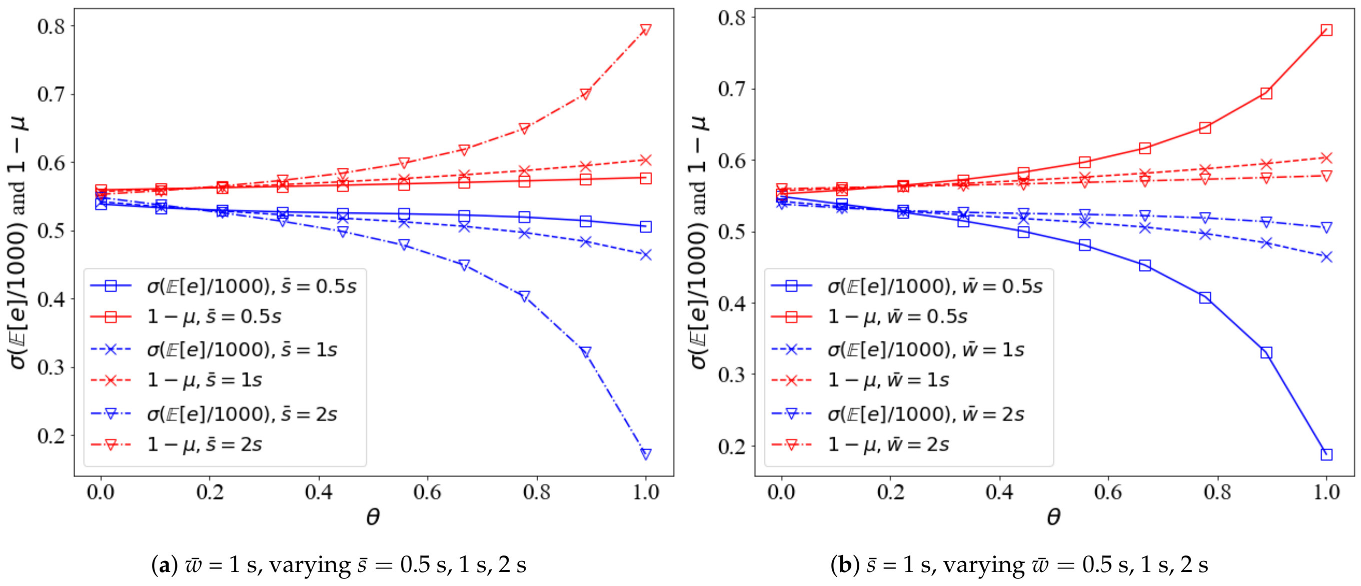
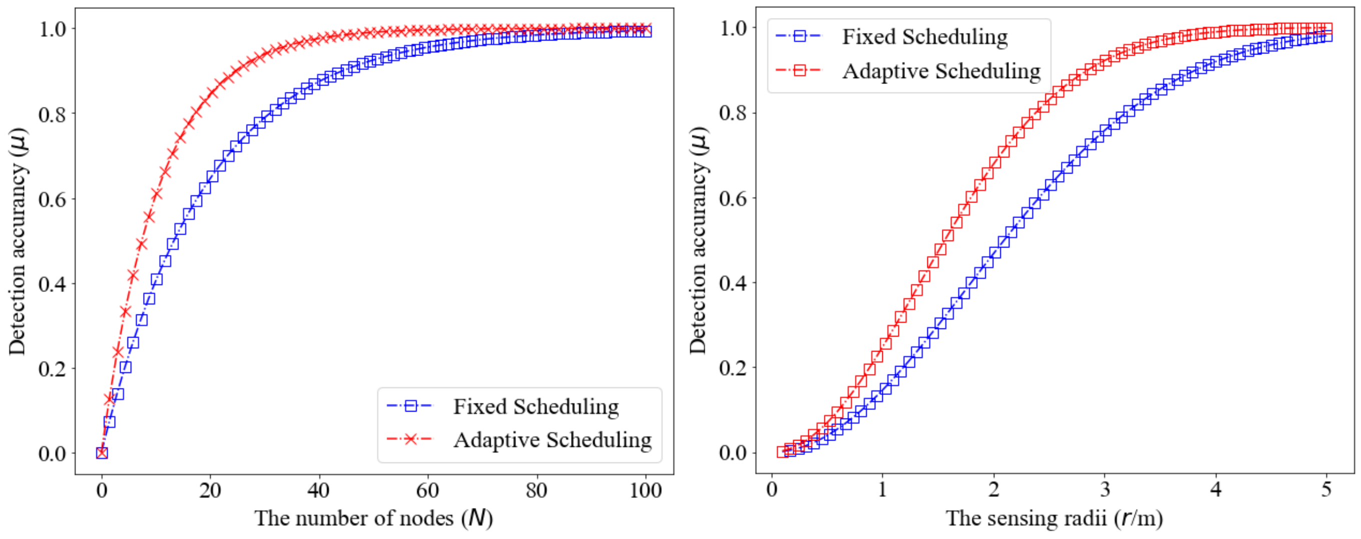


Figure 11. The detection accuracy and energy depletion against θ. (**a**) $\bar{w}$ is fixed to 1 s, varying $\bar{s}$; (**b**) $\bar{s}$ is fixed to 1 s, varying $\bar{w}$.

Next, we set up a benchmark strategy: a fixed sleep time strategy, in which $S_k = \bar{s} = W_k = \bar{w} = 1$ s. The relationship between the number of nodes N, the sensing radius r and the overall detection accuracy is studied. The results are shown in Figure 12. For both the proposed adaptive scheduling and fixed scheduling strategies, the detection accuracy will improve with the increase of the sensing radius and the number of nodes. However, the detection accuracy improvement of the proposed adaptive scheduling strategy is significantly higher than that of the fixed scheduling strategy.



(**a**) μ and N (**b**) μ and r

Figure 12. Detection accuracy for the proposed adaptive scheduling and fixed scheduling (**a**) against the number of cognitive sensors N (r is fixed and $r = 3.15$ m); and (**b**) sensing radius r (N is fixed and $N = 100$).

6.2.3. Optimization

By constructing a 3-D surface from the scattered data (see Figure 13), we found that $J(\delta, \theta)$ is a bivariate multi-modal function, which favors the FTO algorithm.

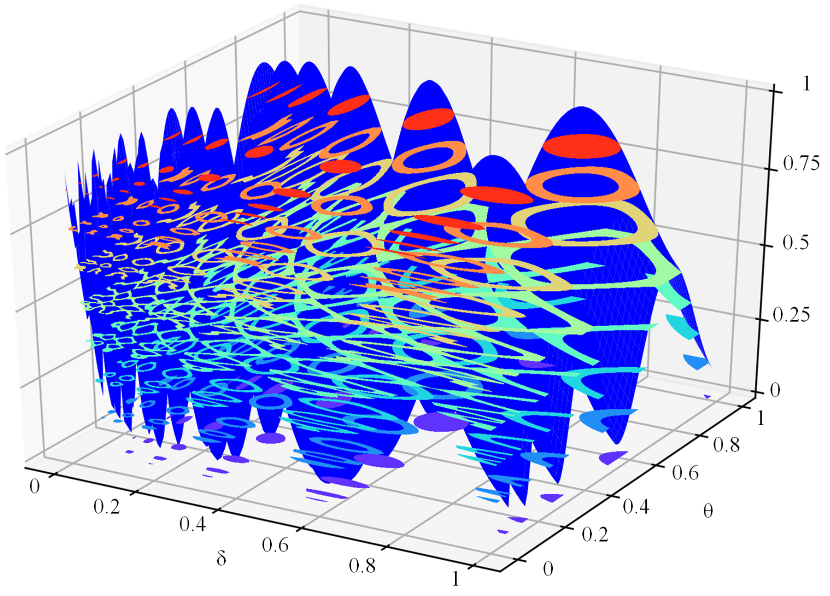
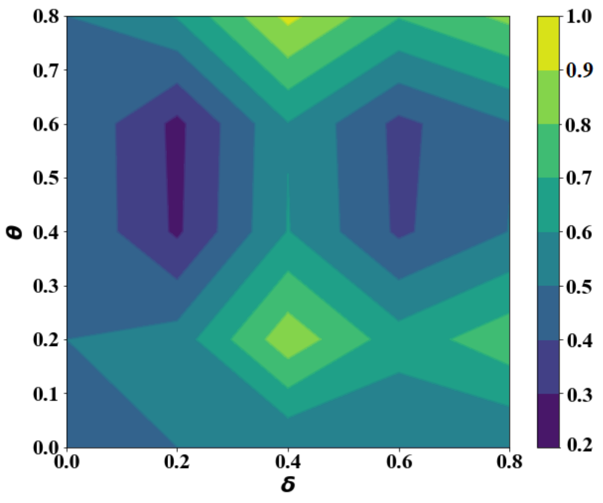
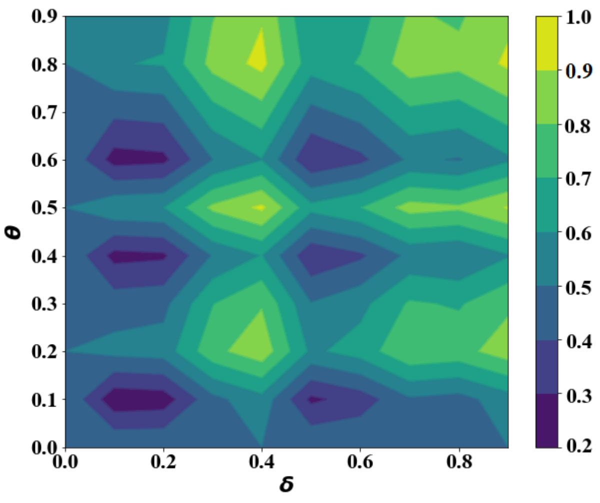


Figure 13. 3-D surface graph for scattered data.

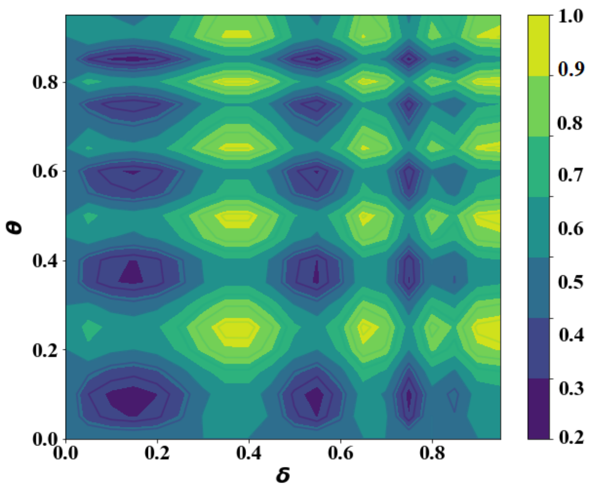
Figure 14 shows the optimization process of the FTO algorithm, and the values of the horizontal and vertical axis represent the values of δ and θ, respectively. The darker the color of the contour map, the smaller the value of J. The algorithm stops after 500 iterations, and Figure 15 provides the final comparison of the peak areas. We can see that the peak area in the lower-left corner is the largest, and is where the optimal point is most likely to appear. The central point of the aforementioned area was taken as the ultimate global optimization point with the coordinate pair $(\delta, \theta) = (0.175, 0.175)$.



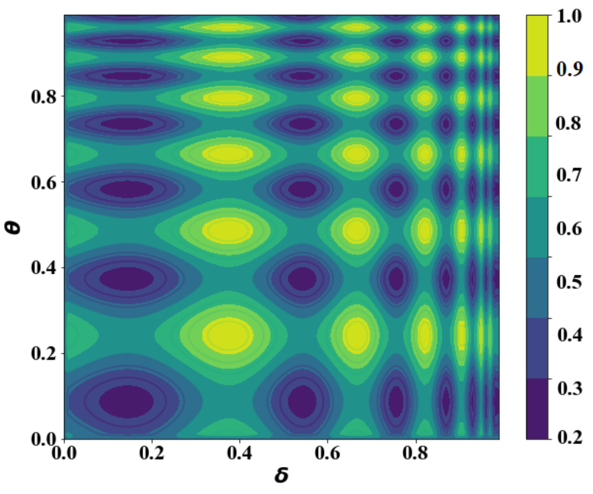
(**a**) 10 iterations



(**b**) 50 iterations



(**c**) 200 iterations



(**d**) 500 iterations

Figure 14. The contour plots in iterations.

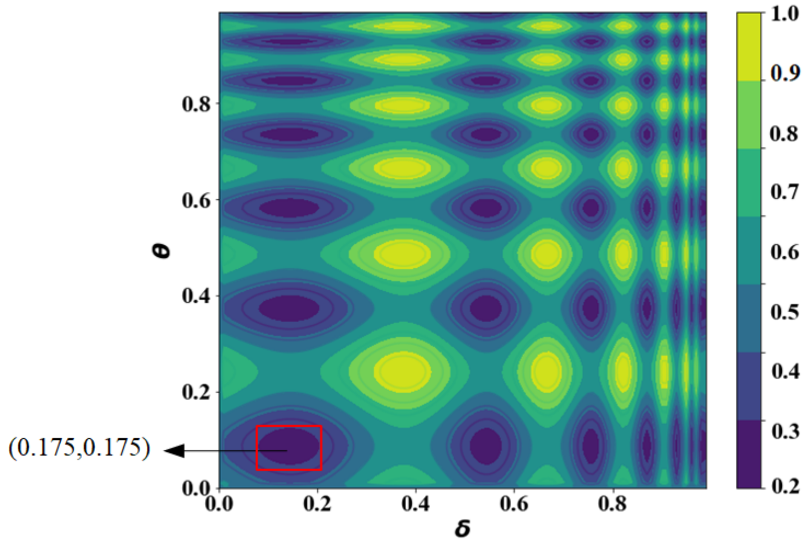


Figure 15. The determination of the ultimate optimum point.

As a comparison, we also implemented the particle swarm optimization (PSO) algorithm, genetic algorithm (GA), comprehensive learning PSO (CLPSO) algorithm, differential evolution (DE) algorithm, and the artificial bee colony (ABC) algorithm. Table 10 outlines the parameters, and Table 11 presents their optimization results. The FTO algorithm outperforms others in most iteration rounds.

Table 10. The algorithm parameters adopted.

Algorithm	Parameter	Value
Particle Swarm Optimization (PSO)	Population	20
	Cognitive ratio	2
	Social coefficient	2
	Inertia weight	0.4~0.9
Genetic Algorithm (GA)	Population	100
	Point mutation	0.01
	Hoist mutation	0.4
	Parsimony coefficient	0.01
Comprehensive Learning Particle Swarm Optimization (CLPSO)	Learning rate	0.05~0.5
	Population	20
	Cognitive ratio	2
	Social coefficient	2
	Inertia weight	0.4~0.9
Differential Evolution (DE)	Population	20
	Scaling factor	0.6
	Crossover rate	0.8
Artificial Bee Colony (ABC)	Followers	50
	Scouters	20
	Employed foragers	1
Fibonacci Tree Optimization (FTO)	Nested branch depth	2
	Total branch depth	6
	Search space	[0,1]
	Max iterations	1000
	Precision	0.001

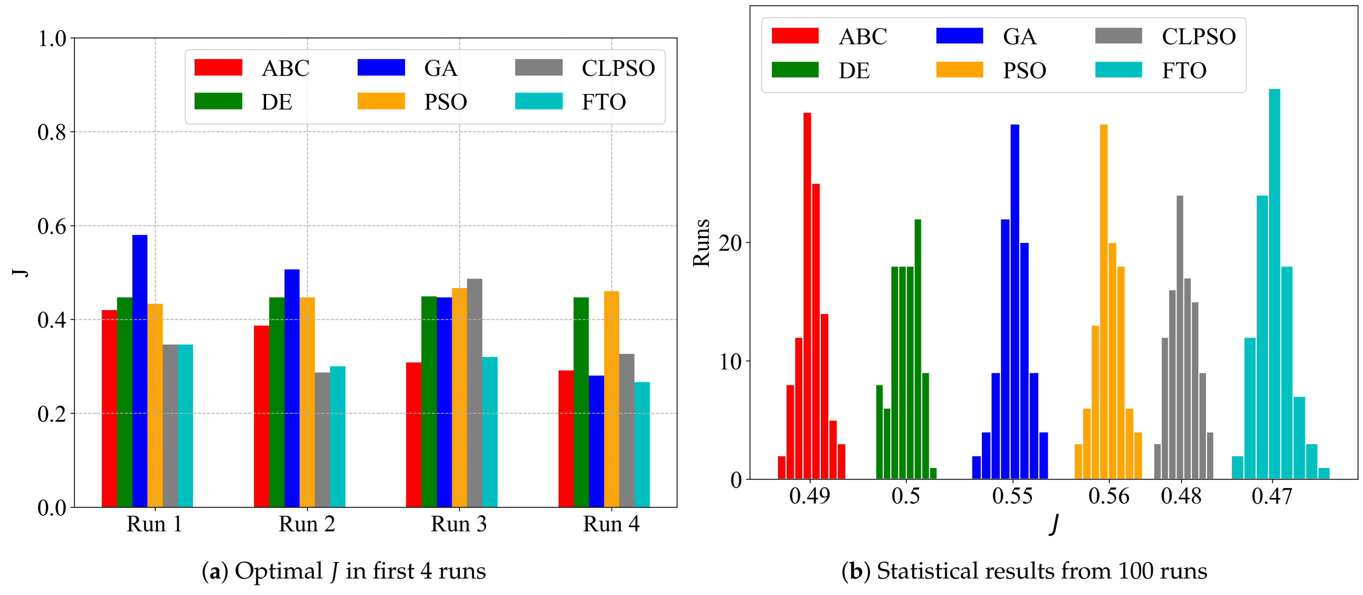
Table 11. The optimization results.

Algorithm	**Iter = 10**			**Iter = 50**			**Iter = 200**			**Iter = 500**		
	min J	**δ**	**θ**	**min J**	**δ**	**θ**	**min J**	**δ**	**θ**	**min J**	**δ**	**θ**
PSO	0.65	0.57	0.165	0.67	0.71	0.11	0.55	0.75	0.34	0.56	0.59	0.65
GA	0.87	0.57	0.2	0.76	0.38	0.6	0.57	0.56	0.13	0.55	0.55	0.65
CLPSO	0.52	0.72	0.4	0.49	0.18	0.74	0.56	0.75	0.9	0.54	0.15	0.15
DE	0.67	0.57	0.37	0.67	0.78	0.56	0.5	0.44	0.15	055	0.65	0.65
ABC	0.63	0.57	0.38	0.58	0.51	0.17	0.49	0.45	0.65	0.49	0.65	0.5
FTO	0.52	0.74	0.56	0.45	0.18	0.86	0.45	0.8	0.13	0.45	0.175	0.175

6.3. Test

Once the parameters of the strategy are determined, they remain unchanged in the coming hours.

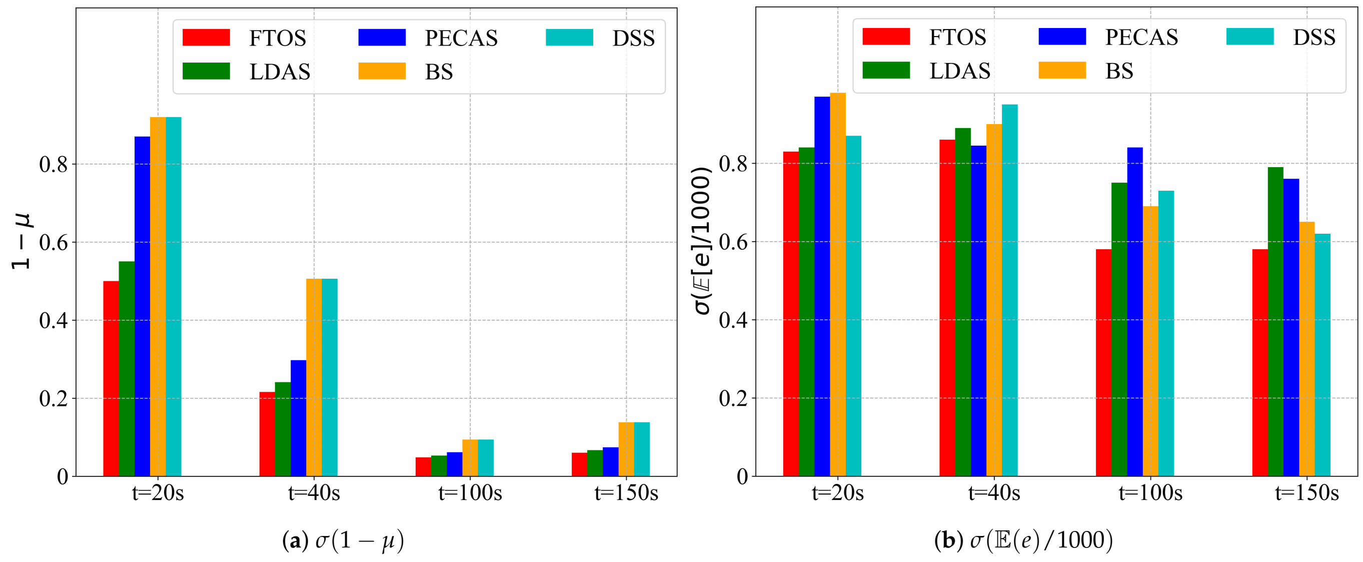
Figure 16a presents the optimal J for the first four runs, applying ABC, DE, GA, PSO, CLPSO and FTO; Figure 16b shows a statistical result from 100 runs. From the perspective of distribution, FTO demonstrates a better optimistic effect than the other algorithms for its lowest center J value 0.47.



(**a**) Optimal J in first 4 runs (**b**) Statistical results from 100 runs

Figure 16. Test results of FTOS scheduling with parameters optimized by different algorithms.

We have implemented several sensing scheduling strategies including LDAS, PECAS, BS and DSS. These strategies all make similar assumptions. Figure 17 shows the results. FTOS ($\delta = \theta = 0.175$) accomplished better detection accuracy and consumes the bare minimum of energy compared to the other sensing scheduling strategies. For sensing duration (t = 20~150 s), all sensing scheduling strategies improved, viz., $1 - \mu$ and $\mathbb{E}[e]$ decline with time.



(**a**) $\sigma(1 - \mu)$ (**b**) $\sigma(\mathbb{E}(e)/1000)$

Figure 17. The expected detection accuracy and the expected energy depletion of different sensing scheduling strategies at different time scales.

7. Deficiencies and Unresolved Issues

The research of this paper is still flawed in the following aspects:

1. Selection of fitting functions. Some simple fitting functions are selected by experience to simulate the relationship between variables related to the objective function and the parameters of the scheduling strategy. The results are indeed good, but other fitting functions are still worth trying.

2. Selection of sensors. The diffuse reflection photoelectric infrared proximity sensor, E3F-DS100P1, has a sensing radius of 3~5 m. If you have the budget to pay, you had better use sensors with a larger sensing radius and more accuracy, for example, laser radar.
3. Too few nodes. Because of the insufficient number of devices we have, we have only configured a 3-node network. Experiments with more nodes would be more convincing.

The dynamics of network traffic were not analyzed. It is also interesting to associate our strategy with other approaches, such as the follow-up of specific targets in [19], which is also a topic for future efforts.

8. Conclusions

The FTOS strategy proposed in this paper addresses the synergy between the energy availability and fault tolerance of event monitoring. The experimental results show that the FTOS strategy contributes to reducing energy depletion and increasing detection accuracy. It outperforms LDAS, BS, DSS and PECAS, which have the same design objectives. The experimental findings show that the step-size parameters optimized by the FTO algorithm are better than those optimized by PSO, GA, and so forth. We also guide the actual deployment. It should be noted that when applied in practice, the FTOS strategy can be generalized only if the step-size can be systematically formulated.

Author Contributions: Conceptualization, L.W.; Data curation, L.W.; Formal analysis, L.W.; Funding acquisition, H.C.; Investigation, L.W.; Methodology, L.W.; Project administration, H.C.; Software, L.W.; Supervision, H.C.; Validation, L.W.; Visualization, L.W.; Writing—original draft, L.W.; Writing—review & editing, H.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant numbers 61832014.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Research data are be obatained by sending an email to wuliangshun@whu.edu.cn.

Acknowledgments: The authors gratefully acknowledge Hailei Duan for providing source codes of some optimization algorithms.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Phan, K.T.; Huynh, P.; Nguyen, D.N.; Ngo, D.T.; Hong, Y.; Le-Ngoc, T. Energy-Efficient Dual-Hop Internet of Things Communications Network with Delay-Outage Constraints. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4892–4903. [CrossRef]
2. Boddu, N.; Boba, V.; Vatambeti, R. A Novel Georouting Potency Based Optimum Spider Monkey Approach for Avoiding Congestion in Energy Efficient Mobile Ad-hoc Network. *Wirel. Pers. Commun.* **2021**. [CrossRef]
3. Sheng, Z.; Mahapatra, C.; Zhu, C.; Leung, V.C.M. Recent Advances in Industrial Wireless Sensor Networks Toward Efficient Management in IoT. *IEEE Access* **2015**, *3*, 622–637. [CrossRef]
4. Solis, F.; Fernández Bocco, Á.; Galetto, A.C.; Passetti, L.; Hueda, M.R.; Reyes, B.T. A 4GS/s 8-bit time-interleaved SAR ADC with an energy-efficient architecture in 130 nm CMOS. *Int. J. Circ. Theory Appl.* **2021**. [CrossRef]
5. Li, H.; Wang, Z.; Wang, H. An energy-efficient power allocation scheme for Massive MIMO systems with imperfect CSI. *Digit. Signal Process.* **2021**, *112*, 102964. [CrossRef]
6. Begum, S.; Wang, S.; Krishnamachari, B.; Helmy, A. ELECTION: Energy-efficient and low-latency scheduling technique for wireless sensor networks. In Proceedings of the 29th Annual IEEE Conference on Local Computer Networks (LCN), Tampa, FL, USA, 16–18 November 2004
7. Dantu, R.; Abbas, K.; O'Neill, M., II; Mikler, A. Data centric modelling of environmental sensor networks. In Proceedings of the IEEE Globecom, Dallas, TX, USA, 29 November–3 December 2004; pp. 447–452
8. Lee, J.; Lee, D.; Kim, J.; Cho, W.; Pajak, J. A dynamic sensing cycle decision scheme for energy efficiency and data reliability in wireless sensor networks. *Lect. Notes Comput. Sci.* **2007**, *4681*, 218–229.
9. Jain, A.; Chang, E.Y. Adaptive sampling for sensor networks. In Proceedings of the International Workshop on Data Management for Sensor Networks, Toronto, ON, Canada, 30 August 2004; Volume 72, pp. 10–16

10. Jothiraj, S.; Balu, S.; Rangaraj, N. An efficient adaptive threshold-based dragonfly optimization model for cooperative spectrum sensing in cognitive radio networks. *Int. J. Commun. Syst.* **2021**, *34*, e4829. [CrossRef]
11. Chen, S.; Shao, D.; Shu, X.; Zhang, C.; Wang, J. FCC-Net: A Full-Coverage Collaborative Network for Weakly Supervised Remote Sensing Object Detection. *Electronics* **2020**, *9*, 1356. [CrossRef]
12. Franceschi, S.; Chirici, G.; Fattorini, L.; Giannetti, F.; Corona, P. Model-assisted estimation of forest attributes exploiting remote sensing information to handle spatial under-coverage. *Spat. Stat.* **2021**, *41*, 100472. [CrossRef]
13. He, T. Energy-efficient surveillance system using wireless sensor networks. In Proceedings of the MobiSys 04, Boston, MA, USA, 6–9 June 2004.
14. Wu, K. Lightweight deployment-aware scheduling for wireless sensor networks. *ACM/Kluwer Mob. Netw. Appl.* **2005**, *10*, 837–852. [CrossRef]
15. Zhang, Y.; Zhang, D.; Vance, N.; Li, Q.; Wang, D. A Light-weight and Quality-aware Online Adaptive Sampling Approach for Streaming Social Sensing in Cloud Computing. In Proceedings of the 24th IEEE International Conference on Parallel and Distributed Systems (ICPADS), Singapore, 11–13 December 2018.
16. Ye, F.; Zhong, G.S.; Zhang, L. PEAS: A robust energy conserving protocol for long-lived sensor networks. In Proceedings of the IEEE International Conference on Network Protocols (ICNP), Paris, France, 12–15 November 2002
17. Liu, C.H.; Zhao, J.; Zhang, H.; Guo, S.; Leung, K.K.; Crowcroft, J. Energy-Efficient Event Detection by Participatory Sensing under Budget Constraints. *IEEE Syst. J.* **2017**, *11*, 2490–2501. [CrossRef]
18. Friderikos, V.; Papadaki, K.; Wisely, D.; Aghvami, H.A. Non-Independent Randomized Rounding for Link Scheduling in Wireless Mesh Networks. In Proceedings of the 64th IEEE Vehicular Technology Conference, Montreal, QC, Canada, 25–28 September 2006.
19. Sun, M. Adaptive Sensing Schedule for Dynamic Spectrum Sharing in Time-Varying Channel. *IEEE Trans. Veh. Technol.* **2018**, *67*, 5520–5524. [CrossRef]
20. Hanef, M.; Deng, Z. Design challenges and comparative analysis of cluster based routing protocols used in Wireless Sensor Networks for improving network lifetime. *Adv. Inf. Sci. Serv. Sci.* **2012**, *4*, 450–459.
21. Alam, K.M.; Kamruzzaman, J.; Karmakar, G.; Murshed, M. Dynamic adjustment of sensing range for event coverage in wireless sensor networks. *J. Netw. Comput. Appl.* **2014**, *46*, 139–153. [CrossRef]
22. Cichon, K.; Kliks, A.; Bogucka, H. Energy-efficient cooperative spectrum sensing: A survey. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1861–1886. [CrossRef]
23. Liu, J.L.; Ravishankar, C.V. LEACH-GA: Genetic Algorithm-based energy efficient adaptive clustering protocol for Wireless Sensor Networks. *Int. J. Mach. Learn. Comput.* **2011**, *1*, 79–85. [CrossRef]
24. Salim, A.; Osamy, W.; Khedr, A.M. IBLEACH: Intra-balanced Leach protocol for Wireless Sensor Networks. *Wirel. Netw.* **2014**, *20*, 1515–1525. [CrossRef]
25. Mishra, S.; Yaduvanshi, R.; Dubey, K.; Rajpoot, P. ESS-IBAA: Efficient, short, and secure ID-based authentication algorithm for wireless sensor network. *Int. J. Commun. Syst.* **2021**, *34*, e4764. [CrossRef]
26. Heo, N.; Varshney, P.K. An intelligent deployment and clustering algorithm for adistributed mobile sensor network. In Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Washington, DC, USA, 5–8 October 2003
27. Song, C. Selective CS: An Energy-Efficient Sensing Architecture for Wireless Implantable Neural Decoding. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2018**, *8*, 201–210. [CrossRef]
28. Sendra, S.; Lloret, J.; Garcia, M.; Toledo, J.F. Power saving and energy optimization techniques for wireless sensor neworks. *J. Commun.* **2011**, *6*, 439–459. [CrossRef]
29. Wang, C.; Song, T.; Wu, J.; Jiang, W.; Hu, J. Energy-Efficient Optimal Sensing and Resource Allocation of Soft Cooperative Spectrum Sensing in CRNs. In Proceedings of the 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xi'an, China, 23–25 October 2019; pp. 1–6
30. Shakkottai, S.; Srikant, R.; Shroff, N. Unreliable sensor grids: Coverage, connectivity, and diameter. In Proceedings of the IEEE INFOCOM, San Francisco, CA, USA, 30 March–3 April 2003
31. Dhumal, S.; Shetty, B.S. Energy-Efficient Coverage and Sensor Localization for Scheduling. In Proceedings of the International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 17–19 July 2019; pp. 537–541
32. Rebiha, S.; Fouzi, S. Energy-efficient coverage and connectivity of wireless sensor network in the framework of hybrid sensor and vehicular network. *Int. J. Comput. Appl.* **2020**. [CrossRef]
33. Holland, J.H. Erratum: Genetic Algorithms and the Optimal Allocation of Trials. *Siam J. Comput.* **2006**, *2*, 88–105. [CrossRef]
34. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
35. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, The University of Western Australia, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948
36. Liang, J.J.; Qin, A.K.; Suganthan, P.N.; Baskar, S. Comprehensive learning particle swarm optimizer for global optimization of multi-modal functions. *IEEE Trans. Evol. Comput.* **2006**, *10*, 281–295. [CrossRef]
37. Zhu, B.; Zhu, F. Discrete artificial bee colony algorithm based on logic operation. *Acta Electron. Sin.* **2015**, *43*, 2161–2166.
38. Wang,Y.; Gao, Y.; Zhou, M.; Yu, Y. A Multi-Layered Gravitational Search Algorithm for Function Optimization and Real-World Problems. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 94–109. [CrossRef]
39. Chen, X.; Jiang, H. Research of quantum tabu search algorithm. *Acta Electron. Sin.* **2013**, *41*, 2161–2166.

40. Das, S.; Maity, S. Real-parameter evolutionary multimodal optimization—A survey of the state of the art. *Swarm Evol. Comput.* **2011**, *2*, 71–78. [CrossRef]
41. Lalbakhsh, P.; Zaeri, B.; Lalbakhsh, A. An improved model of ant colony optimization using a novel pheromone update strategy. *IEICE Trans. Inf. Syst.* **2013**, *96*, 2309–2318. [CrossRef]
42. Cao, Y.; Zhang, H.; Li, W.; Zhou, M.; Zhang, Y.; Chaovalitwongse, W.A. Comprehensive Learning Particle Swarm Optimization Algorithm With Local Search for Multimodal Functions. *IEEE Trans. Evol. Comput.* **2019**, *23*, 718–731. [CrossRef]
43. Jamshidi, M.; Alibeigi, N.; Oryani, B.; Lalbakhsh, A.; Soheyli, M.R.; Rabbani, N. Socialization of Industrial Robots: An Innovative Solution to Improve Productivity. In Proceedings of the IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada, 1–3 November 2018; pp. 832–837
44. Zhang, H.; Zeng, F. A Fibonacci Branch Search (FBS)-Based Optimization Algorithm for Enhanced Nulling Level Control Adaptive Beamforming Techinique. *IEEE Access* **2019**, *7*, 160800–160818. [CrossRef]