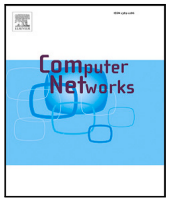
Computer Networks 254 (2024) 110778

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Lifetime maximization of IoT-enabled smart grid applications using error control strategies

Nazli Tekin [a,*], Bilge Kagan Dedeturk [a], Vehbi Cagri Gungor [b]

[a] Department of Software Engineering, Erciyes University, 38280 Kayseri, Turkey
[b] Department of Computer Engineering, Abdullah Gul University, 38080 Kayseri, Turkey

ARTICLE INFO

Keywords:
Internet of things
Error control
Smart grid
Network lifetime

ABSTRACT

Recently, with the advancement of Internet of Things (IoT) technology, IoT-enabled Smart Grid (SG) applications have gained tremendous popularity. Ensuring reliable communication in IoT-based SG applications is challenging due to the harsh channel environment often encountered in the power grid. Error Control (EC) techniques have emerged as a promising solution to enhance reliability. Nevertheless, ensuring network reliability requires a substantial amount of energy consumption. In this paper, we formulate a Mixed Integer Programming (MIP) model which considers the energy dissipation of EC techniques to maximize IoT network lifetime while ensuring the desired level of IoT network reliability. We develop meta-heuristic approaches such as Artificial Bee Colony (ABC) and Particle Swarm Optimization (PSO) to address the high computation complexity of large-scale IoT networks. Performance evaluations indicate that the EC-Node strategy, where each IoT node employs the most energy-efficient EC technique, yields a minimum of 8.9% extended lifetimes compared to the EC-Net strategies, where all IoT nodes employ the same EC method for a communication. Moreover, the PSO algorithm reduces the computational time by 77% while exhibiting a 2.69% network lifetime decrease compared to the optimal solution.

1. Introduction

In recent years, Internet of Things (IoT) technology has gained significant attention in Smart Grid (SG) applications. The integration of IoT technology with SG applications offers several benefits such as improved energy efficiency, reduced costs, and increased reliability [1,2]. Such SG applications include demand response management [3], SG monitoring and control [4], distributed energy resources management [5], and energy management systems [6]. The demand response system allows utilities to reduce energy demand during peak hours by offering incentives to customers who reduce their energy consumption, while SG monitoring and control systems enable utilities to monitor and control power flow in real-time, ensuring efficient power delivery and reducing energy waste. Additionally, distributed energy resources management systems are responsible for managing solar panels and wind turbines, ensuring efficient and reliable power delivery. Furthermore, energy management systems play a crucial role in monitoring energy consumption in real-time and making informed decisions to reduce energy usage and costs.

Besides many tangible benefits offered by IoT-enabled SG applications, there exist challenges in ensuring reliable communication over IoT networks due to the harsh channel environment that SG often encounters. To mitigate these challenges, robust Error Control (EC) techniques, namely, Automatic Repeat Request (ARQ), Forward Error Correction (FEC), and Hybrid ARQ (HARQ) are employed to detect and correct errors for ensuring reliable data transmission in IoT-enabled SG applications [7,8]. However, the utilization of EC techniques to ensure reliability in IoT-enabled SG applications comes at the expense of increased energy consumption, which is critical for the IoT network lifetime.

To address the challenges mentioned above, in this paper, we investigate the performance of EC techniques in terms of energy consumption and their impact on the IoT network lifetime. In addition, we introduce a new node-level EC strategy (*i.e.,* EC-Node) such that each IoT node employs the most energy-efficient EC techniques for each transmission to maximize the IoT network lifetime. We develop Mixed Integer Programming (MIP) to maximize IoT network lifetime as well as ensure network reliability. Finally, we propose meta-heuristic approaches such as the Artificial Bee Colony (ABC) algorithm and the Particle Swarm Optimization (PSO) algorithm to decrease the time complexity of the MIP model.

Contributions. Our main contributions to this study are summarized as follows:

* Corresponding author.
E-mail addresses: nazlitekin@erciyes.edu.tr (N. Tekin), bilgededeturk@erciyes.edu.tr (B.K. Dedeturk), cagri.gungor@agu.edu.tr (V.C. Gungor).

https://doi.org/10.1016/j.comnet.2024.110778
Received 19 June 2023; Received in revised form 11 June 2024; Accepted 3 September 2024
Available online 12 September 2024

Table 1
Literature overview.

Existing works	SG-IoT channel model	Error control & correction	Network lifetime analysis	Network relability
[9]	✗	✓	✗	✗
[10]	✗	✓	✗	✗
[11]	✗	✓	✗	✗
[12]	✗	✓	✗	✓
[13]	✗	✓	✗	✗
[14]	✓	✓	✗	✗
Our work	✓	✓	✓	✓

- We employ a realistic channel model of IoT networks for SG applications, specifically designed to simulate the communication conditions in a 500 kV Line of Sight (LOS) outdoor SG environment.
- We examine the implications of implementing various EC techniques, including ARQ, FEC, and HARQ, on IoT-enabled SG network lifetime, while ensuring the desired level of network reliability.
- We develop a MIP framework for EC-Node strategy to maximize the network lifetime while assuring the desired level of network reliability of IoT-enabled SG applications.
- Due to the NP-hard nature of our MIP model, the time complexity increases exponentially as the size of the IoT networks grows, particularly when dealing with hundreds of IoT nodes. To overcome this challenge and achieve feasible solutions within a reasonable time frame, we introduce two meta-heuristic approaches: Artificial Bee Colony (ABC) and Particle Swarm Optimization (PSO).

Organization. The rest of the paper is organized as follows. Section 2 presents the previous works on IoT-enabled SG applications and EC techniques implementation. elaborates the channel model for IoT networks, energy, and delay cost for EC techniques and presents the MIP model, ABC, and PSO algorithms. Section 5 presents the performance analysis. Section 6 concludes the paper.

2. Related work

In recent years, the use of IoT in SG applications has become a popular research topic in the literature. Several recent surveys about IoT-enabled SG have been conducted. For instance, [15,16] provide a comprehensive review of communication technologies, architectures, and applications of IoT-aided SG systems. [17] presents a survey focused on techniques for maximizing network lifetime in wireless sensor networks.

IoT Network Lifetime. Sarwesh et al. [18] introduce a network architecture designed for energy efficiency and reliability, aimed at enhancing the lifetime of IoT networks. The proposed architecture integrates routing and node placement techniques, strategically adjusting relay sensor node density hierarchies to balance energy consumption. Al Hamadi et al. [19] presents a novel Integer Non-Linear Programming (INLP) optimization model focused on maximizing the lifetime of IoT networks, encompassing both Sigfox star networks and Time Slotted Channel Hopping (TSCH) mesh networks. The proposed model addresses the challenge of balancing reliability and energy consumption by determining the optimal redundancy level in the presence of unreliable sensing environments while achieving the Quality-of-Service (QoS) requirements. Shreyas et al. [20] introduce an Energy Efficient Routing Scheme (EERS) tailored for duty-cycled nodes to maximize the lifetime of IoT networks. EERS employs sleep/awake modes based on residual energy, coverage area, and total active time, optimizing energy consumption. Darabkh et al. [21] introduce an energy-aware cluster-based protocol that employs Particle Swarm Optimization (PSO) for cluster head selection. The protocol features an adaptive mobile sink to ensure balanced energy dissipation among nodes through a dynamic circular path. Batta et al. [22] propose an Improved Lifetime Optimization Clustering (ILCK) approach that employs the Kruskal minimal spanning tree heuristic to address the challenge of lifetime improvement in energy-constrained IoT networks. Yarinezhad et al. [23] develop a new clustering method that employs a 1.2-approximation algorithm to balance data traffic load among cluster heads, addressing the issue of premature node depletion. Additionally, they introduce an energy-aware routing algorithm that distributes communication load efficiently to maximize the network lifetime. Halder et al. [24] propose a Lifetime Maximizing optimal Clustering Algorithm (LiMCA), incorporating a stochastic deployment scheme and a centralized training protocol.

Error Control and Correction in IoT. Tsimbalo et al. [9] evaluate the error correction techniques which employ two iterative decoding algorithms, namely, ADMM and BP to provide energy-efficient communication in IoT networks. Mahapatra et al. [10] present a novel fault detection and error correction scheme for data transmission in IoT networks for smart cities. The proposed scheme utilizes redundant residue arithmetic which offers a low complexity, delay-tolerant, and energy-efficient solution. Han et al. [11] propose a novel cross-layer communication module to address diverse device and service characteristics of IoT networks. They provide an optimization model to minimize packet error rate, energy consumption, and delay. Yigit et al. [12] analyze error control and correction schemes for the SG environment, and develop an adaptive error control method that leverages RS codes with OQPSK modulation for SG applications. Triphati et al. [13] introduce a channel-adaptive data transmission model in SG IoT applications based on the channel conditions which are predicted by using the stochastic method and data learning method. The prediction method is combined with an adaptive channel coding scheme to enhance the reliability of transmitting time-critical grid monitoring data over wireless channels. Naoor-A-Rahim et al. [14] introduce an IoT-based SG system to monitor the status of microgrids over a wireless network. The system employs a delay-universal-based error correction code to ensure reliable and real-time estimation of microgrids. They also present the iterative estimation technique leveraging the distinctive characteristics of the delay-universal code. (see Table 1).

Differences from other works. Most of the above-mentioned research studies primarily concentrate on the design of an energy-efficient routing algorithm to maximize network lifetime. A limited number of studies investigate the EC for IoT networks. To the best of our knowledge, no previous studies consider realistic channel conditions of SG and the tradeoffs between network reliability and energy efficiency for IoT-enabled SG applications. This is the first study that focuses on maximizing IoT-enabled SG network lifetime while ensuring the desired level of network reliability. In this paper, we provide a network lifetime analysis of the IoT-enabled SG networks by formulating the MIP model that considers energy dissipation of various EC techniques. Furthermore, we develop meta-heuristics to address the time complexity challenges of large-scale IoT network scenarios.

3. Background

In this section, we give a brief description of EC methods. Error control methods are fundamentally classified into three main approaches:

Table 2
Nomenclature used in this paper.

Symbol	Description	Unit	Value	Symbol	Description	Unit	Value
$\overline{\text{PL}_{ij}}$	Path loss bw. node-i and node-j	dB	–	$T^c_{tx,ij}$	Transmission time for $c \in \{ARQ, FEC, HARQ\}$	s	–
$\overline{\text{PL}_0}$	Reference path loss [25]	dB	63.57	$T^c_{rx,ij}$	Reception time for $c \in \{ARQ, FEC, HARQ\}$	s	–
d_{ij}	Distance bw. node-i and node-j	m	–	T_{acq}	Duration to acquire data [26]	ms	20
d_0	Reference distance [27]	m	0.5	T_{dec}	Duration to decode FEC code	s	–
η	Path loss exponent [25]	–	2.42	T_{grd}	Guard time	s	–
σ	Shadowing standard deviation [25]	dB	3.12	T^p_{ij}	Propagation delay	s	–
$\overline{\zeta_{ij}}$	Signal-to-noise ratio (SNR)	dB	–	T_{rnd}	Round duration	s	600
$\overline{P_n}$	Noise floor [28]	dBm	–93	P_{tx}	Transmission power [29]	mW	5
$\overline{P_t}$	Output power [29]	dBm	0	P_{rx}	Reception power [26]	mW	21.4
BER_{ij}	Bit error rate bw. node-i and node-j	–	–	P_{slp}	Sleep power [29]	μW	3
BLER_{ij}	Block error rate bw. node-i and node-j	–	–	P_{std}	Standby power [29]	mW	35.4
B_N	Noise bandwidth [28]	kHz	30	c	Speed of light	m/s	3×10^8
R	Data rate [28]	kbps	19.2	l^c_{pck}	Packet size for $c \in \{ARQ, FEC\}$	byte	–
PER^c_{ij}	Packet error rate for $c \in \{ARQ, FEC, HARQ\}$	dB	7.35	l_{pl}	Payload size	byte	128
μ^c_{ji}	Packet success rate at node-j for $c \in \{ARQ, FEC, HARQ\}$	–	–	l_a	ACK packet size [26]	byte	20
$n^c_{\text{ret},ij}$	Expected number of retransmissions for $c \in \{ARQ, FEC, HARQ\}$	–	–	l_h	Header size [26]	byte	12
$E^c_{tx,ij}$	Transmission energy cost for $c \in \{ARQ, FEC, HARQ\}$	J	–	N_R	Network lifetime	rounds	–
$E^c_{rx,ji}$	Reception energy cost for $c \in \{ARQ, FEC, HARQ\}$	J	–	$p_{gen,i}$	Amount of generated packets at each round	–	–
E_{enc}	Encoding energy cost [30]	J	≈ 0	p^c_{ij}	Number of packets flows for $c \in \{ARQ, FEC, HARQ\}$	–	–
E_{dec}	Decoding energy cost	J	–	ψ	Target PDR	–	0.5–0.999
E_{ini}	Initial battery energy [26]	kJ	25	$\mathcal{N}$	Set of all IoT nodes	–	100–220
E^{to}_{ij}	Energy consumed before timeout	J	–	$\mathcal{C}$	Set of EC methods	–	–
E_{acq}	Data acquisition energy [26]	μJ	600	R_{net}	Network radius	m	20

Automatic Repeat Request (ARQ), Forward Error Correction (FEC), and Hybrid Automatic Repeat Request (HARQ).

ARQ: ARQ employs acknowledgments and timeouts to ensure reliable data transfer. If no error is detected in transmitted data, the receiver informs transmitters with a positive acknowledgment (ACK). Conversely, if there is an erroneous or lost packet in transmitted data, the receiver informs the transmitter with a negative acknowledgment (NACK) and requests retransmissions. Timeout is a period for the transmitter to wait for an acknowledgment. If the transmitter does not receive an ACK or NACK packet before timeout, it retransmits the packet until it receives acknowledgment or exceeds the number of retransmissions.

FEC: FEC adds redundant bits to the transmitted data packets, allowing the receiver to detect and correct errors without requiring the sender to resend the data. Bose-Chaudhuri-Hocquenghem (BCH) and Reed Solomon (RS) are the commonly used FEC codes in communication systems. These codes denoted as (n, k, t), determines the code's capabilities of correcting errors. Here, n is the total block length, k is the length of the payload (*i.e., original message*), and t represents the number of errors the code can correct.

HARQ: HARQ is a hybrid approach that leverages both ARQ and FEC methods. If errors are detected, the method first attempts error correction using FEC. If FEC alone is insufficient, automatic retransmission is invoked. There are two types, namely HARQ-I and HARQ-II. In HARQ-I, the transmitter sends the data packet and waits for an acknowledgment. If a Negative Acknowledgment (NACK) is received, the transmitter resends the data, this time encoded with FEC. In HARQ-II, if a NACK packet is received, the transmitter resends only the redundant bits of the data packet.

4. System model

In this section, we first introduce the channel model. Subsequently, we provide energy and delay cost models of different EC methods. Later, we present the MIP framework. Finally, we define our meta-heuristic approaches. Table 2 provides the nomenclature used in this paper.

4.1. Channel model

We employ a realistic channel model for IoT networks to better represent signal propagation characteristics [28,31]. This model allows us to capture the path loss between IoT nodes by considering various factors such as distance, obstacles, and environmental conditions. The experimental studies have substantiated its superiority over Nakagami and Rayleigh models in accurately representing multipath channels within wireless environments [32,33]. The path loss between node-i and node-j is determined by

$$\overline{\text{PL}_{ij}} = \overline{\text{PL}_0} + 10\eta \log_{10}(d_{ij}/d_0) + \overline{X_\sigma}, \tag{1}$$

where d_{ij}, d_0, $\overline{\text{PL}_0}$ stand for the distance between node-i and node-j, the distance and path loss at the reference point, respectively. η is the path loss exponent and $\overline{X_\sigma}$ is a zero mean Gaussian random variable with the standard deviation σ, respectively.

For a given packet length of q bits, the packet error rate (PER) for ARQ and FEC are calculated as follows [34]:

$$\text{PER}^{\text{ARQ}}_{ij}(q) = 1 - (1 - \text{BER}_{ij})^q, \tag{2}$$

$$\text{PER}^{\text{FEC}}_{ij}(q) = 1 - (1 - \text{BLER}_{ij})^{\lceil \frac{q}{k} \rceil}. \tag{3}$$

The bit error rate (BER) and block error rate (BLER) for FEC block code represented as (n, k, t) where n is the block length, k is the payload length and t is the error correcting capability are given by [28]

$$\text{BER}_{ij} = 0.5 \exp\left(-0.5 \times 10^{0.1 \times \overline{\zeta_{ij}}} \times (B_N/R)\right), \tag{4}$$

$$\text{BLER}_{ij} = \sum_{l=t+1}^{n} \binom{n}{l} (\text{BER}_{ij})^l (1 - \text{BER}_{ij})^{n-l}, \tag{5}$$

where B_N and R denote the noise bandwidth and data rate, respectively. $\lceil \frac{q}{k} \rceil$ is the number of blocks required to send q bits. The signal-to-noise ratio (SNR) [28] is given by

$$\overline{\zeta_{ij}} = \overline{P_t} - \overline{\text{PL}_{ij}} - \overline{P_n}, \tag{6}$$

where $\overline{P_t}$ and $\overline{P_n}$ are the output power and the noise floor, respectively. Therefore, the packet success rates (PSR) for ARQ, FEC, and HARQ are given by

$$\begin{aligned}\mu_{ji}^{\mathrm{ARQ}} &= 1-\left[\mathrm{PER}_{ji}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\times \mathrm{PER}_{ji}^{\mathrm{ARQ}}(l_a)\right]^{\left(n_{\mathrm{ret},ji}^{\mathrm{ARQ}}+1\right)},\\ \mu_{ji}^{\mathrm{FEC}} &= 1-\mathrm{PER}_{ji}^{\mathrm{FEC}}\left(l_{pck}^{\mathrm{FEC}}\right),\\ \mu_{ji}^{\mathrm{HARQ}} &= 1-\left[\mathrm{PER}_{ji}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\times \mathrm{PER}_{ji}^{\mathrm{FEC}}\left(l_{pck}^{\mathrm{FEC}}\right)\right],\end{aligned} \tag{7}$$

respectively. l_{pck}^{ARQ} and l_{pck}^{FEC} are packet size for ARQ and FEC, respectively. l_a denotes acknowledgment (ACK) packet size. $n_{\mathrm{ret},ij}^{\mathrm{ARQ}}$ is the number of retransmissions calculated as [26]

$$n_{\mathrm{ret},ij}^{\mathrm{ARQ}} = \left[\left(1-\mathrm{PER}_{ij}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\right)\times\left(1-\mathrm{PER}_{ji}^{\mathrm{ARQ}}\left(l_a\right)\right)\right]^{-1}. \tag{8}$$

4.2. Communication model costs

Communication energy refers to the energy consumption involved in both the transmission and reception of a data packet. In ARQ, the necessity to retransmit all data packets up to the maximum allowable number of transmissions imposes a substantial energy cost on the IoT node. The energy consumed for data transmission is determined by [26]

$$E_{tx,ij}^{\mathrm{ARQ}} = E_{enc} + n_{\mathrm{ret},ij}^{\mathrm{ARQ}}\times\left[E_{tx}\left(l_{pck}^{\mathrm{ARQ}}\right)+E_{rx}(l_a)+E_{ij}^{to}\right], \tag{9}$$

where E_{enc} is the encoding energy cost which is negligible small. $E_{tx}\left(l_{pck}^{\mathrm{ARQ}}\right)$ is the transmission energy cost of a packet size $l_{pck}^{\mathrm{ARQ}} = l_{pl}+l_h$ and $E_{rx}(l_a)$ is the energy cost for receiving ACK packet. Here, l_{pl}, l_h, and l_a represent the payload size, header size, and ACK packet size, respectively. E_{ij}^{to} refers to the timeout energy consumed by the sender during the waiting period for acknowledgment and calculated as:

$$\begin{aligned}E_{ij}^{to} &= P_{std}\times(T_{ij}^{p}+T_{grd}),\\ T_{ij}^{p} &= 2\times d_{ij}/c,\\ T_{grd} &= 2\times\max_{i,j} T_{ij}^{p},\end{aligned} \tag{10}$$

where P_{std} is the standby power. T_{ij}^{p}, T_{grd} and c are the round trip propagation delay, the guard time and the speed of light in vacuum, respectively.

The energy required for transmitting and receiving q bits data are expressed as follows:

$$\begin{aligned}E_{tx}(q) &= qE_{elec}+P_{tx}(q/R),\\ E_{rx}(q) &= qE_{elec}+P_{rx}(q/R),\end{aligned} \tag{11}$$

where P_{tx} and P_{rx} are transmission power and reception power, respectively. The energy consumed for data reception is calculated as:

$$E_{rx,ji}^{\mathrm{ARQ}} = n_{\mathrm{ret},ji}^{\mathrm{ARQ}}\times\left[E_{rx}\left(l_{pck}^{\mathrm{ARQ}}\right)+E_{tx}(l_a)\right]. \tag{12}$$

In FEC, redundancy bits are added to the data packets before the transmission. These redundancy bits result in an extra energy cost for both the transmitter and the receiver. Moreover, the decoding process incurs energy costs for the receiver. Therefore, the size of an FEC packet is given by $l_{pck}^{\mathrm{FEC}} = \lceil\frac{l_{pl}}{k}\rceil n + l_h$. The transmission and reception energy costs are calculated as:

$$E_{tx,ij}^{\mathrm{FEC}} = E_{enc}+E_{tx}\left(l_{pck}^{\mathrm{FEC}}\right), \tag{13}$$

$$E_{rx,ji}^{\mathrm{FEC}} = E_{rx}\left(l_{pck}^{\mathrm{FEC}}\right)+E_{dec}, \tag{14}$$

respectively. $E_{dec} = P_{proc}T_{dec}$ is the energy consumed for decoding an FEC packet where P_{proc} is processing power and T_{dec} is decoding time defined in Section 4.3.

In HARQ-I, when a NACK is received after transmitting a data packet with ARQ, the transmitter retransmits the encoded data packet using FEC. The transmitter incurs additional energy consumption during retransmission, while the receiver faces additional energy costs associated with the decoding process. On the other hand, in HARQ-II, when the NACK packet is received, the sender node retransmits only redundant bits to recover the lost data. Therefore, for HARQ-I the packet size in retransmission is given by $\ell = l_{pck}^{\mathrm{FEC}}$, whereas for HARQ-II, it is given by $\ell = \lceil\frac{l_{pl}}{k}\rceil(n-k)+l_h$. The transmission and reception energy costs are calculated as:

$$\begin{aligned}E_{tx,ij}^{\mathrm{HARQ}} &= E_{enc}+E_{tx}\left(l_{pck}^{\mathrm{ARQ}}\right)+\mathrm{PER}_{ij}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\\ &\quad\times\left[E_{rx}(l_{na})+E_{tx}(\ell)\right],\end{aligned} \tag{15}$$

$$\begin{aligned}E_{rx,ji}^{\mathrm{HARQ}} &= E_{rx}\left(l_{pck}^{\mathrm{ARQ}}\right)+\mathrm{PER}_{ji}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\\ &\quad\times\left[E_{tx}(l_{na})+E_{rx}(\ell)+E_{dec}\right],\end{aligned} \tag{16}$$

respectively where l_{na} is the size of negative acknowledgment packet.

4.3. Delay model costs

In ARQ, the overall transmission time comprises the time required to send a data packet, including the propagation delay and a guard time for waiting to receive an acknowledgment (ACK) packet. Conversely, the reception time refers to the time taken to receive the data packet. Consequently, the calculations of transmission and reception times for ARQ are given by [35]

$$\begin{aligned}T_{tx,ij}^{\mathrm{ARQ}} &= \left(l_{pck}^{\mathrm{ARQ}}/R\right)+T_{ij}^{p}+T_{grd},\\ T_{rx}^{\mathrm{ARQ}} &= \left(l_{pck}^{\mathrm{ARQ}}/R\right).\end{aligned} \tag{17}$$

In FEC, the transmission time comprises the duration required to transmit a data packet along with the propagation delay, as there is no requirement to wait for an acknowledgment (ACK) packet. Conversely, the reception time encompasses the time spent on receiving the data packet as well as the time required for decoding the received packet. The time for transmitting and receiving data can be expressed as: [35]

$$\begin{aligned}T_{tx,ij}^{\mathrm{FEC}} &= \left(l_{pck}^{\mathrm{FEC}}/R\right)+T_{ij}^{p},\\ T_{rx}^{\mathrm{FEC}} &= \left(l_{pck}^{\mathrm{FEC}}/R\right)+T_{dec},\end{aligned} \tag{18}$$

respectively. The time spent for decoding an FEC packet that consists of $\lceil l_{pl}/k\rceil$ number of FEC blocks is calculated as follows:

$$\begin{aligned}T_{dec} &= T^{blck}dec\times\lceil l_{pl}/k\rceil,\\ T_{dec}^{blck} &= (2nt+2t^2)\times(T_{add}+T_{mult}).\end{aligned} \tag{19}$$

T_{dec}^{blck} is the time required for decoding an FEC block code (n,k,t). T_{add} and T_{mult} denote the execution times for addition and multiplication operations in the central processing unit (CPU) of the microcontroller. Given that an 8-bit microcontroller performs addition and multiplication in 1 and 2 cycles, respectively, $T_{add}+T_{mult}$ is determined by $3\frac{\lceil log_2 n+1\rceil}{8}t_{cycle}$ where t_{cycle} is the one cycle period [36].

In HARQ, the initial transmission is carried out using ARQ, while the subsequent retransmission utilizes an FEC code. Thus, the time duration for transmitting and receiving data are expressed as [35] :

$$\begin{aligned}T_{tx,ij}^{\mathrm{HARQ}} &= T_{tx,ij}^{\mathrm{ARQ}}+\mathrm{PER}_{ij}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\times T_{tx,ij}^{\mathrm{FEC}},\\ T_{rx,ji}^{\mathrm{HARQ}} &= T_{rx}^{\mathrm{ARQ}}+\mathrm{PER}_{ji}^{\mathrm{ARQ}}\left(l_{pck}^{\mathrm{ARQ}}\right)\times T_{rx}^{\mathrm{FEC}}.\end{aligned} \tag{20}$$

4.4. MIP framework

In this section, we introduce a disk-shaped network topology. The network comprises IoT nodes including the sink node (*i.e.,* node-1) which is positioned in the center. The set, $\mathcal{N}$, denotes all IoT nodes in the network. Note that $\mathcal{N}\backslash\{1\}$ indicates all IoT nodes except the sink node. Each IoT node in the network continuously senses collects data, and transmits it to the sink node. It is worth noting that the sink node does not produce or transmit any data packets, and it does not have any

constraints on energy. The network lifetime refers to the time until the battery of the first IoT node in the network is depleted. The network lifetime is structured into rounds, with the pre-defined duration of each round denoted as T_{rnd} and the total number of rounds defined as N_R. The sink node collects the generated data at the end of each round.

We present a MIP framework designed to maximize the IoT network lifetime through the utilization of distinct EC strategies. We define two types of strategies, namely EC-Net and EC-Node. In EC-Net, a homogeneous approach is adopted where all IoT nodes employ the same EC method for a communication. The framework for EC-Net strategy is presented in our previous work [26]. Conversely, in EC-Node, each IoT node selects a different EC method in each data communication. We define set C as a set of different EC methods composed of ARQ, BCH(31,11,5), BCH(31,21,5), RS(15,11,2), HARQ-I, HARQ-II. The decision variables, parameters, and scalars used in the MIP framework are listed below:

Decision Variables:

- $p^c_{i,j}$ represents the number of packet flows over the node-i to node-j by using one of the EC methods where $c \in C$.
- $T_{bsy,i}$ represents the cumulative duration that the node-i is engaged in packet transmission, reception, and acquisition throughout its lifetime.

Parameters:

- μ^c_{ij} represents the packet success rate over the node-i to node-j by using one of the EC methods where $c \in C$.
- $p_{gen,i}$ represents the number of data packets generated by each node-i except the sink node at each round.
- $E^c_{tx,ij}$ and $E^c_{rx,ij}$ represent the energy consumed for the transmission and reception of a packet from node-i to node-j by using one of the EC methods where $c \in C$, respectively.
- $T^c_{tx,ij}$ and $T^c_{rx,ij}$ represent the time spent for the transmission and reception of a packet from node-i to node-j by using one of the EC methods where $c \in C$, respectively.
- $n^c_{ret,ij}$ represents a packet retransmission by using one of the EC methods where $c \in C$.

Scalars:

- ψ represents the target packet delivery ratio at the sink node.
- T_{acq} represents the time spent for a packet acquisition.
- T_{rnd} represents the duration of each round.
- E_{acq} represents the energy consumed for a packet acquisition.
- E_{tot} represent total energy of the initial battery.
- ς represents the total network bandwidth.
- P_{slp} represents the sleep power.

The MIP model is subject to several constraints as defined in Eq. (21). Eq. (21a) presents the objective of the MIP framework which is to maximize the IoT network lifetime. Eq. (21b) specifies no negative flows in the network. Eq. (21c) guarantees a balance in the flow of packets at each sensor node-i. Eq. (21d) ensures the target packet delivery ratio (*i.e.,* ψ) at the sink node. Eq. (21e) and (21f) ensure that the total busy time of any IoT nodes is less than the network lifetime. Eq. (21g) limits that total energy consumed for acquisition, transmission, reception, and sleep for each node-i is less than the initial battery. Finally, Eq. (21h) restricts the packet transmission and reception flow according to the available network bandwidth.

4.5. Meta-heuristics

In this section, we present two meta-heuristic algorithms, namely ABC and PSO to optimize the MIP model for the EC-Node strategy developed in the previous subsection. The aim is to provide optimal solutions for large-scale IoT networks in a reasonable amount of time. Note that in the EC-Node strategy, the objective is to determine the optimal error control (EC) method employed by each IoT node to maximize the network lifetime. Here, we leverage meta-heuristic approaches to explore solution space (*i.e.,* determining which node should use which EC method) and the MIP model to compute the network lifetime (*i.e.,* incorporating flow variables for routing). We found that the IoT nodes employ primarily HARQ-II, HARQ-I and BCH(31,21,5) methods to ensure the desired level of network reliability. Given that HARQ-II offers better energy efficiency compared to HARQ-I, and BCH(31,21,5) has been selected as the FEC method for HARQ, we limit the use of HARQ-II and BCH(31,21,5) to simplify the decision-making process for each IoT node. This restriction reduces computational complexity, allowing for a solution within an acceptable time. Therefore, the problem becomes identifying which nodes should use HARQ-II or BCH(31,21,5) to maximize the IoT network lifetime. We introduce a solution vector, denoted as $\mathbb{V} = [s_2, s_3, \dots, s_{|\mathcal{N}|}]$, to use as a baseline for meta-heuristic approaches. In this vector, each s_i (where i corresponds to a specific IoT node) is represented by either 0 or 1. Specifically, $s_i = 1$, when node-i utilizes HARQ-II, and $s_i = 0$, when node-i utilizes BCH(31,21,5). The size of the vector corresponds to the total number of IoT nodes (*i.e.,* $|\mathcal{N}|$) in the network.

$$\text{Maximize } N_R \tag{21a}$$

subject to:

$$p^c_{ij} \geq 0, \forall i \in \mathcal{N}\backslash\{1\}, \forall j \in \mathcal{N}, \forall c \in C \tag{21b}$$

$$\sum_{c\in C}\sum_{j\in\mathcal{N}} p^c_{ij} - \sum_{c\in C}\sum_{j\in\mathcal{N}\backslash\{1\}} p^c_{ji}\mu^c_{ji} = N_R \times p_{gen,i}, \forall i \in \mathcal{N}\backslash\{1\} \tag{21c}$$

$$\sum_{c\in C}\sum_{j\in\mathcal{N}\backslash\{1\}} p^c_{j1}\mu^c_{j1} \geq \left(\sum_{i\in\mathcal{N}\backslash\{1\}} p_{gen,i}\right) \times N_R \times \psi, \tag{21d}$$

$$T_{bsy,i} = \sum_{c\in C}\left(\sum_{j\in\mathcal{N}} n^c_{\text{ret},ij} T^c_{tx,ij} p^c_{ij} + \sum_{j\in\mathcal{N}\backslash\{1\}} n^c_{\text{ret},ji} T^c_{rx,ji} p^c_{ji}\right) + N_R \times T_{acq}, \forall i \in \mathcal{N}\backslash\{1\} \tag{21e}$$

$$T_{bsy,i} \leq N_R \times T_{rnd}, \forall i \in \mathcal{N}\backslash\{1\} \tag{21f}$$

$$\sum_{c\in C}\left(\sum_{j\in\mathcal{N}} E^c_{tx,ij} p^c_{ij} + \sum_{j\in\mathcal{N}\backslash\{1\}} E^c_{rx,ji} p^c_{ji}\right) + N_R \times E_{acq} + P_{slp} \times \left(N_R \times T_{rnd} - T_{bsy,i}\right) \leq E_{tot}, \forall i \in \mathcal{N}\backslash\{1\} \tag{21g}$$

$$l^c_{pck} \times \left(\sum_{j\in\mathcal{N}\backslash\{1\}} p^c_{ij} n^c_{\text{ret},ij} + \sum_{j\in\mathcal{N}\backslash\{1\}} p^c_{ji} n^c_{\text{ret},ji}\right) \leq \varsigma \times N_R \times T_{rnd} \forall i \in \mathcal{N} \tag{21h}$$

4.5.1. Artificial Bee Colony (ABC) algorithm

The ABC algorithm is a population-based optimization method that iteratively improves the solution by simulating the food source exploitation and the waggle dance communication behaviors of the bees. The ABC algorithm employs a population of bees consisting of three distinct types: employed bees, onlooker bees, and scout bees. Each type of bee performs a specific function in the search process. Employed bees exploit the food sources in their local regions and generate candidate solutions. Onlooker bees select the most promising solutions from the employed bees and follow them. Scout bees are responsible for exploring new regions by randomly generating new solutions, thus, ensuring the diversity of the population.

The Alg. 1 presents the pseudo-code for the ABC algorithm. In the context of the ABC algorithm, the solution vector is referred to as a food source represented by $\mathcal{F} = [s_2, \dots, s_{|\mathcal{N}|}]$. Note that an IoT node employs either HARQ-II or BCH(31,21,5) denoted by 1 and 0, respectively. The algorithm requires the number of food sources (`fs`), maximum cycle (`maxCycle`), limit (`lim`), lower bound (`lb`), upper

bound (ub), and modification rate (mr) as input parameters. The ABC algorithm starts with an initial population of randomly generated food sources represented as $\mathcal{P} = \{\mathcal{F}_1 = [0,1,0,0,\ldots] \vdots \mathcal{F}_{\mathtt{fs}} = [1,1,0,1,\ldots]\}$ (step 1). The algorithm evaluates the lifetime of each food source by using the MIP model given in Section 4.4 in the initial population. Here, the algorithm tries to find the optimal solution vector to maximize the IoT network lifetime. Since the search space is reduced by the ABC algorithm, we can solve (21a) in a reasonable time. Later, the algorithm memorizes the best lifetime (step 2). During the employed bees phase of the ABC algorithm, each solution in the population undergoes a local search to improve its quality (step 3). Specifically, a new food source is generated under the modification rate condition by

$$v_{ij} = \begin{cases} x_{ij} + \phi_{ij}(x_{ij} - x_{kj}), & \text{if } r_{ij} < mr \\ x_{ij}, & \text{otherwise.} \end{cases} \tag{22}$$

Here, v_{ij} represents the jth sensor node of the new food source, which is generated for the ith employed bee. x_{ij} represents the jth sensor node of the current food source associated with the employed bee, while x_{kj} represents the corresponding sensor node of randomly selected food source in the population that is different from the current food source. The parameters r_{ij} and ϕ_{ij} are the random numbers within the range $[0,1]$ and $[-1,1]$, respectively. After that, the ABC algorithm evaluates the lifetimes of the newly generated food sources and employs a greedy selection strategy to choose the better ones. Finally, if the new food sources cannot provide better lifetimes, the limit counters for each food source are increased. The next step is to compute the probability values for each food source by

$$prob_i = \frac{f_i}{\sum_{n=1}^{fs} f_n} \tag{23}$$

where f_i is a fitness function of ith food source which is the network lifetime in this case (step 4). The onlooker bees phase then selects food sources based on their fitness value (quality) using a probabilistic selection method and tries to improve these selected food sources for getting better lifetimes (step 5). The best lifetime in the current population is memorized (step 6). Finally, the scout bees phase introduces new solutions to the population by randomly generating new food sources if the counter reaches the limit value (step 7). The algorithm repeats these three phases until the maximum cycle is reached (step 8) and returns the best lifetime (step 9).

4.5.2. Particle Swarm Optimization (PSO) algorithm

Particle Swarm Optimization (PSO) is a computational optimization algorithm inspired by the collective behavior of social animals such as birds and fish. The algorithm is based on a population of particles, each representing a potential solution to the optimization problem. The particles move through the solution space and communicate with each other to share information about their best solution found so far. This information exchange allows the particles to collectively explore the search space and converge to the optimal solution. The movement of each particle is influenced by its own experience and the experience of its neighbors, and the algorithm is guided by a set of mathematical formulas that update the velocity and position of each particle in each iteration.

Alg. 2 presents the pseudo-code for PSO algorithm. In the context of the PSO algorithm, the solution vector is referred to as a particle represented by $\mathcal{R} = [s_2, \ldots, s_{|\mathcal{N}|}]$. Note that an IoT node employs either HARQ-II or BCH(31,21,5) denoted by 1 and 0, respectively.. The algorithm requires the number of particles (np) in a population, inertia weight (w), cognitive parameter (c_1) and social parameter (c_2) to control the influence of personal and social experiences on each particle movement, maximum cycle (maxCycle), lower bound (lb) and upper bound (up) as input parameters. Initially, the algorithm generates a population with random particles ($\mathcal{P} = \{\mathcal{R}_1 = [0,1,0,0,\ldots] \vdots \mathcal{R}_{\mathtt{np}} = [1,1,0,1,\ldots]\}$) and their velocity ($v$) (step 1). Later, the algorithm computes the lifetimes of each particle by using the MIP model and

Algorithm 1 The ABC algorithm.

Input: The number of food source (fs ← 10), Maximum cycle (maxCyc ← 10), Limit (lim ← 10), Lower bound (lb ← 0), Upper bound (ub ← 1), Modification rate (mr ← 0.2) .

Output: Network Lifetime

1: Initialize solution population with random food sources ($\mathcal{P} \leftarrow \{\mathcal{F}_1 = [0,1,0,0,\ldots] \vdots \mathcal{F}_{\mathtt{fs}} = [1,1,0,1,\ldots]\}$ where $\mathcal{F}_i$), set cycle ← 0 and counter (counter ← 0)

2: Evaluate the best lifetime from food sources in the population (bestLT ← bestLT(P)).

3: Apply the employed bee phase for a local search.
Generate a new food source by using the Eq. (22)
Evaluate the lifetime. Perform greedy selection and save the best lifetime
If the new lifetime could not be improved, the limit counter increases (counter++), reset the counter (counter ← 0) otherwise.

4: Compute the probability values for each food source based on their quality by using the Eq. (23)

5: Apply the onlooker bee phase to choose food sources based on the probability values. Set a counter $t \leftarrow 0$.
REPEAT Line 4 **IF** random > $prob_i$ for each food sources **UNTILL** $t \leftarrow fs$

6: Memorize the best lifetime (bestLT)

7: Apply the scout bee phase to generate a new food source if an exhausted one exists in the population.
IF (counter > lim) **THEN** Generate a new random food source

8: **REPEAT** Step 3-12 **UNTILL** cycle = maxCycle.

9: **Result:** bestLT

stores them in $\mathtt{PbestLT}_i$ (step 2). The algorithm saves the best lifetime in the population into GbestLT (step 3. The next step is to update the velocity and position of each particle $\mathcal{R}_i$ for searching for better solutions by using the following equations

$$\begin{aligned} v_{new,i} = \mathtt{w} * v_i + \mathtt{c}_1 * \mathtt{r}_1 * (x_{GbestLT} - x_i) \\ + \mathtt{c}_2 * \mathtt{r}_2 * (x_{PbestLT_i} - x_i), \end{aligned} \tag{24}$$

$$x_{new,i} = x_i + v_{new,i} \tag{25}$$

where $\mathtt{r}_1$ and $\mathtt{r}_2$ indicate random number in the range of $[0,1]$ (step 4). Update the local best lifetimes ($\mathtt{PbestLT}_i$) of each particle if the new ones are greater than the old one (step 5) and update the global best lifetime (GbestLT) in the population (step 6). Finally, the algorithm repeats these steps until the stopping criteria and returns the best lifetime (steps 7 and 8).

5. Performance analysis

In this section, we present performance analysis results. We utilize General Algebraic Modeling System (GAMS) and MATLAB to solve both the MIP framework and meta-heuristic approaches. We perform our simulations HP Z640 Workstation with 16 GB RAM and an Intel(R) Xeon(R) E5-2620 v3 processor. We generated disk-shaped connected network topologies with a fixed radius of 20 meters where IoT nodes are randomly deployed. A connected network is defined such that each

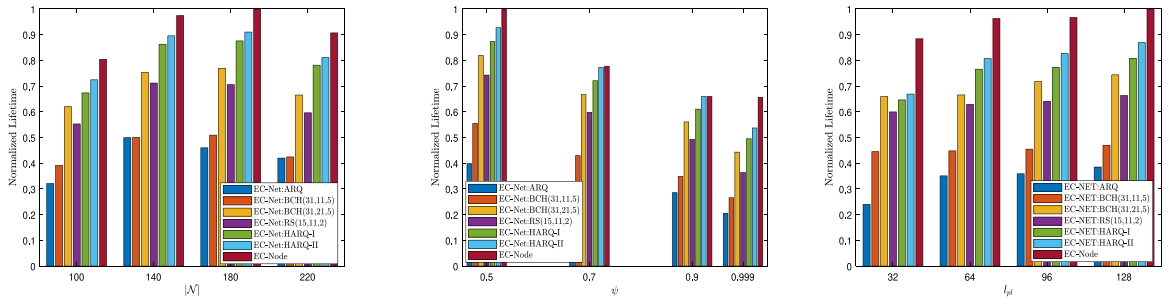


Fig. 1. Normalized lifetimes comparison of EC-Node strategy and EC-Net strategies.

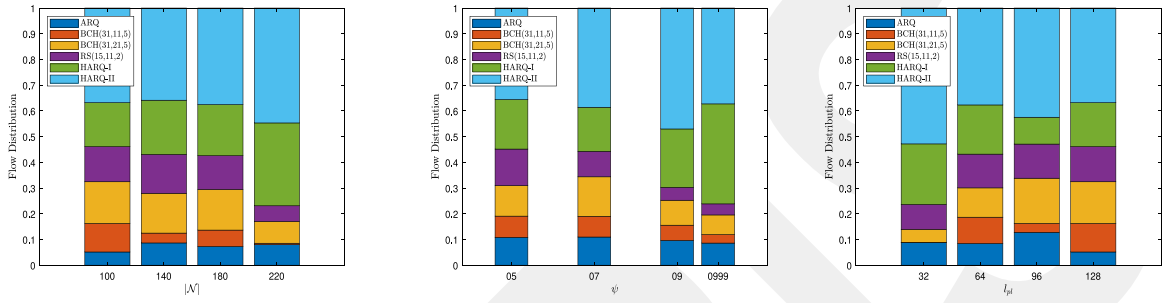


Fig. 2. Flow distribution of EC method usage in EC-Node strategy.

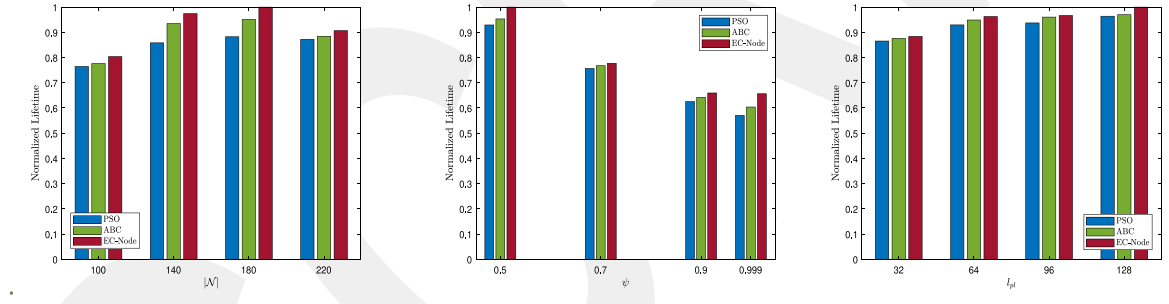


Fig. 3. Normalized lifetimes comparison of EC-Node against meta-heuristics.

node has at least one neighbor, ensuring that the distance between connected nodes is smaller than 3 m. It is crucial to emphasize that our analyses rely on the averages obtained from 20 randomly generated scenarios, considering the stochastic nature of path loss values that take on random values. To accurately evaluate our performance analysis, we provide a normalized lifetime calculated as each data point divided by the maximum data point in each figure. We performed simulations with 500 kV LOS outdoor SG environment and the channel parameters are given in Table 2 [25].

Throughout the analysis, we choose $|\mathcal{N}| = 100$, $\psi = 0.8$, and $l_{pl} = 128$. We systematically examine the impacts of varying numbers of IoT nodes ($|\mathcal{N}|$), target PDR (ψ), and payload size (l_{pl}). In Fig. 1(a), we present the comparison of the EC-Node strategy and EC-Net strategies with respect to the number of IoT nodes. As the number of IoT nodes increases the normalized lifetimes for all strategies exhibit an upward trend due to a reduction in hop distance as the network density increases. Consequently, IoT nodes expend less energy for communication, leading to an enhancement in lifetimes. However, after a certain threshold (*i.e.*, $|\mathcal{N}| = 180$), the normalized lifetimes exhibit a decrease. The observed result can be attributed to the exponential increase of the generated packets with the increase in the number of IoT nodes resulting in higher energy consumption in the network. Moreover, EC-Node strategy outperforms EC-Net strategies with a minimum improvement of 8.9%. It is followed by HARQ-II, HARQ-I, BCH(31,21,5), RS(15,11,2), BCH(31,11,5), and ARQ, respectively.

The ARQ exhibits the shortest lifetime due to energy consumed for retransmissions to achieve targeted PDR. BCH(31,21,5) demonstrates superior performance in maximizing lifetime compared to BCH(31,11,5). Despite both being capable of correcting the same number of bits, their different payload sizes in the FEC block play a crucial role. The smaller payload size in BCH(31,11,5) results in a higher number of FEC blocks with redundant bits, causing excessive energy consumption during transmission. Additionally, these redundant bits overload the battery of receiver nodes during data decoding. While BCH(31,11,5) and RS(15,11,2) share the same payload size, BCH(31,11,5) offers higher error correction capabilities due to the increased number of redundant bits, consequently leading to higher energy consumption. In other words, the larger FEC block length in BCH(31,11,5) contributes to more energy consumption. As a result, the network lifetime using RS(15,11,2) surpasses that of BCH(31,11,5). Finally, HARQ-II attains an extended lifetime in comparison to HARQ-I since HARQ-II retransmits only redundant bits leads less energy consumption.

Algorithm 2 The PSO algorithm.

Input: The number of particles ($\texttt{np} \leftarrow 10$), Inertia weight ($\texttt{w} \leftarrow 0.6$), Cognitive parameter ($\texttt{c}_1 \leftarrow 1.8$), Social parameter ($\texttt{c}_1 \leftarrow 1.8$), Maximum cycle ($\texttt{maxCyc} \leftarrow 10$), Lower bound ($\texttt{lb} \leftarrow 0$), Upper bound ($\texttt{ub} \leftarrow 1$).

Output: Network Lifetime

1: Initialize solution population with random particles ($\mathcal{P} \leftarrow \{\mathcal{R}_1 = [0,1,0,0,\ldots] \ : \ \mathcal{R}_{\texttt{np}} = [1,1,0,1,\ldots]\}$), and randomly initialize the velocity of particles (v) set $\texttt{cycle} \leftarrow 0$

2: Compute the lifetimes for each particle in the population. Find the personal best lifetime of each particle $\mathcal{R}_i$ and set each to $\texttt{PbestLT}_i$.

3: Find the global best lifetime of the particle in the population and set it to $\texttt{GbestLT} = max(\texttt{PbestLT}_i), 1 < i \leq np$

4: Update the velocity and position of each particle $\mathcal{R}_i$. Compute the new particle's lifetime and set it to $\texttt{pLT}_i$

5: **IF** $\texttt{pLT}_i > \texttt{PbestLT}_i$ **THEN** $\texttt{PbestLT}_i \leftarrow \texttt{pLT}_i$

6: **IF** $\texttt{pLT}_i > \texttt{GbestLT}$ **THEN** $\texttt{GbestLT} \leftarrow \texttt{pLT}_i$

7: **REPEAT** Step 3-6 **UNTILL** $\texttt{cycle} = \texttt{maxCycle}$.

8: **Result:** bestLT

In Figs. 1(b) and 1(c), we present the comparison of the EC-Node strategy and EC-Net strategies with respect to the target PDR and payload size, respectively. As the target PDR increases, the normalized lifetimes of all EC strategies decrease due to the increased energy requirements of IoT nodes to attain the target PDR. However, once a certain threshold is reached (*i.e.,* $\psi = 0.9$), we observe that the EC-Node strategy achieves the target PDR criterion while maintaining a relatively constant energy consumption level. Increasing the payload size decreases the number of packets needed for data transmission. Consequently, this reduces energy consumption in packet overhead, leading to an increase in network lifetime. In summary, the tight reliability criterion primarily affects the IoT network lifetime.

In Figs. 2(a)–2(c), we present the packet flow distribution of the EC methods across the IoT network. We calculate flow distribution as the number of packet transmissions for each EC method, denoted as $\sum_{i\in\mathcal{N}\setminus 1}\sum_{j\in\mathcal{N}} p_{ij}^c$, divided by the total number of packet flows over the network, represented as $\sum_{c\in C}\sum_{i\in\mathcal{N}\setminus 1}\sum_{j\in\mathcal{N}} p_{ij}^c$. The number of packet flows with HARQ-II and HARQ-I is greater because IoT nodes closer to the sink node, which predominantly use these EC methods, have a higher volume of packet flows. As the number of IoT nodes increases, forming a denser network, the utilization of HARQ-I rises due to reduced inter-node distances. This reduction in distance leads to a lower BER, thereby ensuring target reliability with fewer retransmissions and consequently, lower energy consumption. Furthermore, when there is a stringent reliability criterion, such as $\psi = 0.999$, the usage of HARQ-I and HARQ-II increases significantly to maximize the IoT network's lifetime. As the payload size increases, IoT nodes tend to utilize BCH(31,21,5) more than HARQ-II to maximize network lifetime. This contributed to BCH(31,21,5) reduces the need for energy-intensive retransmissions. On the other hand, as the number of IoT nodes increases, forming a denser network, the utilization of HARQ-I rises due to reduced inter-node distances. This reduction in distance leads to a lower BER, thereby ensuring target reliability with fewer retransmissions and consequently, lower energy consumption.

On the other hand, we observe that IoT nodes close to the sink node predominantly utilize HARQ-II, while those located farther away opt for BCH(31,21,5). This allocation can be attributed to the energy-intensive nature of HARQ-II in multi-hop communication scenarios, prompting nodes further from the sink to opt for the more energy-efficient BCH(31,21,5) method to conserve energy and maintain reliable communication over longer distances.

Table 3
Computation times (s) of the EC-Node strategy, ABC, and PSO with respect to the number of IoT nodes.

$\lvert\mathcal{N}\rvert$	Computation times (s)			Lifetime differences (%)	
	EC-Node	ABC	PSO	ABC	PSO
100	16	315	140	3.42	4.88
140	127	509	254	4.03	11.89
180	511	887	454	8.58	11.76
220	3063	1440	709	2.44	2.69

Since our MIP model is an NP-hard problem, the computational complexity is inherently high. Moreover, as the number of IoT nodes in the network increases, the computational complexity grows rapidly, making feasible solutions to the optimization problem challenging. Therefore, we develop meta-heuristic approaches to address the computational complexity and to provide practical solutions. In Fig. 3(a), we compare the normalized lifetimes of EC-Node strategy with meta-heuristic approaches (*i.e.,* ABC and PSO) with respect to the number of IoT nodes. We observe a similar trend depicted in Fig. 1(a), the normalized lifetimes increase until reaching a certain threshold (*i.e.,* $\lvert\mathcal{N}\rvert = 180$) due to similar factors. ABC achieves a better normalized lifetime compared to PSO. The reason for that can be the superior local search capability of ABC and its ability to perform an effective global search, which further enhances its ability to explore the search space and find optimal solutions or near-optimal solutions.

In Fig. 3(b), we present the comparison of EC-Node strategy and meta-heuristic approaches with respect to target PDR in terms of normalized lifetimes. As the target PDR increases, the normalized lifetimes of EC-Node, ABC, and PSO decrease. When the target PDR is too high (*i.e.,* $\psi = 0.999$), the lifetime difference between EC-Node and meta-heuristic approaches namely ABC and PSO, becomes more significant, with the difference of 8.3% and 14.03% respectively. In Fig. 3(c), we present the comparison of EC-Node strategy and meta-heuristic approaches with respect to payload size in terms of normalized lifetimes. We show that increasing payload size extends the network lifetime slightly for all approaches.

Table 3 illustrates the computation times for EC-Node strategy and meta-heuristic approaches as well as the lifetime differences between them with respect to the number of IoT nodes. As the number of IoT nodes increases in the network, the computation times for the EC-Node strategy increase drastically after a certain point. On the other hand, ABC and PSO demonstrate a smoother increase in computation times compared to the EC-Node strategy. Moreover, the lifetime differences are in the range from 2.44% to 11.89% between the EC-Node strategy and meta-heuristic approaches. Specifically, when $\lvert\mathcal{N}\rvert = 220$, ABC provides 2.44% less lifetime with 53% fewer solution times whereas PSO provides 2.69% less lifetime with 77% fewer solution times.

6. Conclusion

In conclusion, the integration of IoT technology with SG applications has brought numerous benefits such as improved energy efficiency, reduced costs, and increased reliability of energy grid. However, ensuring reliable communication over IoT networks in the harsh channel environment of SG poses challenges. To address these challenges, EC techniques such as ARQ, FEC, and HARQ are employed to detect and correct errors in data transmission. While these techniques enhance reliability, they come at the cost of increased energy consumption, which can significantly impact the lifetime of IoT networks.

In this study, we investigated the performance of EC techniques in terms of energy consumption and their impact on the lifetime of IoT networks. We introduced a new node-level EC strategy, EC-Node, where each IoT node utilizes the most energy-efficient EC techniques for each transmission, aiming to maximize the network lifetime. To achieve this, we developed a Mixed Integer Programming (MIP) framework that maximizes the network lifetime of IoT-enabled SG applications while ensuring the network reliability. Furthermore, to address the time complexity challenges associated with large-scale IoT networks, we proposed meta-heuristic approaches, including the ABC and PSO algorithms.

The primary findings of our study can be summarized as follows:

1. The EC-Node strategy demonstrates the highest normalized lifetimes compared to EC-Net strategies. EC-Node strategy outperforms the second-best normalized lifetime, HARQ-II, by achieving a minimum improvement of 8.9%.
2. The meta-heuristic approaches such as ABC and PSO yield normalized lifetimes with reductions ranging from 2.44% to 8.58% and 2.69% to 11.89%, respectively compared to the EC-Node strategy.
3. After a certain threshold (*i.e.,* $\psi = 0.9$, the EC-Node strategy demonstrates fulfilled the desired level of reliability without significant increments in energy consumption.
4. The ABC and PSO yield sub-optimal solution at most with 55% and 77% less time, respectively when the number of IoT nodes in the network is equal to 220.

As part of future work, we intend to integrate energy harvesting systems into the IoT-enabled smart grid environment to assess their impact on network lifetime. Furthermore, we will explore the use of advanced optimization algorithms based on machine learning techniques to address the challenges posed by large-scale IoT networks.

CRediT authorship contribution statement

Nazli Tekin: Writing – review & editing, Writing – original draft, Software, Methodology, Conceptualization. **Bilge Kagan Dedeturk:** Writing – review & editing, Software. **Vehbi Cagri Gungor:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] A. Meloni, P.A. Pegoraro, L. Atzori, A. Benigni, S. Sulis, Cloud-based IoT solution for state estimation in smart grids: Exploiting virtualization and edge-intelligence technologies, Comput. Netw. 130 (2018) 156–165.

[2] E. Fadel, V.C. Gungor, L. Nassef, N. Akkari, M.A. Malik, S. Almasri, I.F. Akyildiz, A survey on wireless sensor networks for smart grid, Comput. Commun. 71 (2015) 22–33.

[3] W.-T. Li, C. Yuen, N.U. Hassan, W. Tushar, C.-K. Wen, K.L. Wood, K. Hu, X. Liu, Demand response management for residential smart grid: From theory to practice, IEEE Access 3 (2015) 2431–2440.

[4] W. Li, T. Logenthiran, V.-T. Phan, W.L. Woo, Implemented IoT-based self-learning home management system (SHMS) for Singapore, Internet Things J. 5 (3) (2018) 2212–2219.

[5] S. Salinas, M. Li, P. Li, Y. Fu, Dynamic energy management for the smart grid with distributed energy resources, IEEE Trans. Smart Grid 4 (4) (2013) 2139–2151.

[6] M.F. Zia, E. Elbouchikhi, M. Benbouzid, Microgrids energy management systems: A critical review on methods, solutions, and prospects, Appl. Energy 222 (2018) 1033–1055.

[7] B.E. Bilgin, V.C. Gungor, Adaptive error control in wireless sensor networks under harsh smart grid environments, Sensor Rev. 32 (3) (2012) 203–211.

[8] M. Yigit, V.C. Gungor, P. Boluk, Performance analysis of hamming code for WSN-based smart grid applications, Turk. J. Electr. Eng. Comput. Sci. 26 (1) (2018) 125–137.

[9] E. Tsimbalo, X. Fafoutis, R.J. Piechocki, CRC error correction in IoT applications, IEEE Trans. Ind. Inform. 13 (1) (2016) 361–369.

[10] C. Mahapatra, Z. Sheng, V.C. Leung, T. Stouraitis, A reliable and energy-efficient IoT data transmission scheme for smart cities based on redundant residue-based error correction coding, in: 12th Annual IEEE International Conference on Sensing, Communication, and Networking-Workshops, SECON Workshops, IEEE, 2015, pp. 1–6.

[11] C. Han, J.M. Jornet, E. Fadel, I.F. Akyildiz, A cross-layer communication module for the Internet of Things, Comput. Netw. 57 (3) (2013) 622–633.

[12] M. Yigit, P.S. Boluk, V.C. Gungor, A new efficient error control algorithm for wireless sensor networks in smart grid, Comp. Stand. Inter. 63 (2019) 27–42.

[13] S. Tripathi, S. De, Channel-adaptive transmission protocols for smart grid IoT communication, IEEE Internet Things J. 7 (8) (2020) 7823–7835.

[14] M. Noor-A-Rahim, M.O. Khyam, M.A. Mahmud, M.T.I. ul Huque, X. Li, D. Pesch, A.M. Oo, Robust and real-time state estimation of unstable microgrids over IoT networks, IEEE Syst. J. 15 (2) (2020) 2176–2185.

[15] Y. Saleem, N. Crespi, M.H. Rehmani, R. Copeland, Internet of Things-aided smart grid: technologies, architectures, applications, prototypes, and future research directions, IEEE Access 7 (2019) 62962–63003.

[16] M.O. Qays, I. Ahmad, A. Abu-Siada, M.L. Hossain, F. Yasmin, Key communication technologies, applications, protocols, and future guides for IoT-assisted smart grid systems: A review, Energy Rep. 9 (2023) 2440–2452.

[17] H. Yetgin, K.T.K. Cheung, M. El-Hajjar, L.H. Hanzo, A survey of network lifetime maximization techniques in wireless sensor networks, IEEE Commun. Surv. Tut. 19 (2) (2017) 828–854.

[18] P. Sarwesh, N.S.V. Shet, K. Chandrasekaran, Energy efficient and reliable network design to improve the lifetime of low power IoT networks, in: 2017 International Conference on Wireless Communications, Signal Processing and Networking, WISPNET, IEEE, 2017, pp. 117–122.

[19] H. Al-Hamadi, M. Saoud, R. Chen, J.-H. Cho, Optimizing the lifetime of IoT-based star and mesh networks, IEEE Access 8 (2020) 63090–63105.

[20] J. Shreyas, H. Deepa, P. Udayaprasad, D. Chouhon, N. Srinidhi, D.K. SM, Energy optimization to extend network lifetime for IoT-based wireless sensor networks, in: 2022 4th International Conference on Smart Systems and Inventive Technology, ICSSIT, IEEE, 2022, pp. 90–93.

[21] K.A. Darabkh, B.A. Asma'a, M. Al-Akhras, K.K. Wafa'a, Improving network lifetime in IoT sensor network based on particle swarm optimization, clustering, and mobile sink, in: 2022 4th Middle East and North Africa COMMunications Conference, MENACOMM, IEEE, 2022, pp. 95–99.

[22] M.S. Batta, Z. Aliouat, H. Mabed, M. Merah, An improved lifetime optimization clustering using Kruskal's mst and batteries aging for IoT networks, in: 2022 International Symposium on Networks, Computers and Communications, ISNCC, IEEE, 2022, pp. 1–6.

[23] R. Yarinezhad, M. Sabaei, An optimal cluster-based routing algorithm for lifetime maximization ofInternet of Things, J. Parallel Distribut. Comput. 156 (2021) 7–24.

[24] S. Halder, A. Ghosal, M. Conti, LiMCA: an optimal clustering algorithm for lifetime maximization of Internet of Things, Wirel. Netw. 25 (2019) 4459–4477.

[25] S. Kurt, H.U. Yildiz, M. Yigit, B. Tavli, V.C. Gungor, Packet size optimization in wireless sensor networks for smart grid applications, IEEE Trans. Ind. Electron. 64 (3) (2016) 2392–2401.

[26] N. Tekin, V.C. Gungor, The impact of error control schemes on lifetime of energy harvesting wireless sensor networks in industrial environments, Comp. Stand. Inter. 70 (2020) 103417.

[27] E. Tanghe, W. Joseph, L. Verloock, L. Martens, H. Capoen, K.V. Herwegen, W. Vantomme, The industrial indoor channel: large-scale and temporal fading at 900, 2400, and 5200 MHz, IEEE Trans. Wirel. Commun. 7 (7) (2008) 2740–2751.

[28] M. Zuniga, B. Krishnamachari, Analyzing the transitional region in low power wireless links, in: Proc. Ann. IEEE Commun. Society Conf. Sens. Ad Hoc Commun. Netw., SECON, 2004, pp. 517–526.

[29] J. Vales-Alonso, E. Egea-López, A. Martínez-Sala, P. Pavón-Mariño, M. Victoria Bueno-Delgado, J. García-Haro, Performance evaluation of MAC transmission power control in wireless sensor networks, Comput. Netw. 51 (6) (2007) 1483–1498.

[30] A. Xenakis, F. Foukalas, G. Stamoulis, Cross-layer energy-aware topology control through simulated annealing for WSNs, Comput. Elec. Eng. 56 (2016) 576–590.

[31] N. Tekin, H.E. Erdem, V.C. Gungor, Analyzing lifetime of energy harvesting wireless multimedia sensor nodes in industrial environments, Comput. Stand. Interfaces 58 (2018) 109–117.

[32] M.Z. Zamalloa, B. Krishnamachari, An analysis of unreliability and asymmetry in low-power wireless links, ACM Trans. Sensor Netw. 3 (2) (2007) 7–es.

[33] V.C. Gungor, B. Lu, G.P. Hancke, Opportunities and challenges of wireless sensor networks in smart grid, IEEE Trans. Ind. Electron. 57 (10) (2010) 3557–3564.
[34] N. Tekin, V.C. Gungor, Analysis of compressive sensing and energy harvesting for wireless multimedia sensor networks, Ad Hoc Netw. 103 (2020) 102164.
[35] N. Tekin, H.U. Yildiz, V.C. Gungor, Node-level error control strategies for prolonging the lifetime of wireless sensor networks, IEEE Sens. J. 21 (13) (2021) 15386–15397.
[36] M.C. Vuran, I.F. Akyildiz, Error control in wireless sensor networks: A cross layer analysis, IEEE/ACM Trans. Netw. 17 (4) (2009) 1186–1199.

Nazli Tekin received her B.S. degree in Computer Engineering from Koç University, Istanbul, Turkey, in 2011, and her Ph.D. degree in Electrical and Computer Engineering from Abdullah Gül University, Kayseri, Turkey, in 2020. In 2021, she was a post-doctoral associate at the Cyber-Physical Systems Security Lab in the Department of Electrical and Computer Engineering at Florida International University, Miami, FL, USA. She was supported by the Scientific and Technological Research Council of Turkey (TUBITAK-BIDEB) 2219—International Postdoctoral Research Scholarship Program. She is currently an Assistant Professor in the Department of Software Engineering at Erciyes University, Kayseri, Turkey. Her research interests include wireless sensor networks and IoT security.

Bilge Kagan Dedeturk received his B.S. degree in Computer Systems Teaching from Gazi University in Ankara, Turkey, in 2010. He earned his M.S. and Ph.D. degrees in Electrical and Computer Engineering from Erciyes University in Kayseri, Turkey, in 2014 and 2020, respectively. He is currently an Assistant Professor in the Department of Software Engineering at Erciyes University. His research interests include machine learning, natural language processing, swarm intelligence, and metaheuristics.

Vehbi Cagri Gungor received his Ph.D. degree in electrical and computer engineering from the Broadband and Wireless Networking Laboratory, Georgia Institute of Technology, Atlanta, GA, USA, in 2007. Currently, he is a Full Professor and Dean of Faculty of Computer Science, Abdullah Gul University in Kayseri, Turkey. His current research interests are in machine learning, machine-to-machine communications, next-generation wireless networks, wireless ad hoc and sensor networks. Dr. Gungor has authored more than 100 papers in refereed journals and international conference proceedings, and has been serving as an editor, reviewer and program committee member to numerous journals and conferences in these areas. He is also the recipient of TUBITAK Young Scientist Award in 2017, Science Academy Young Scientist Award (BAGEP) in 2017, Turkish Academy of Sciences Distinguished Young Scientist Award (TUBA-GEBIP) in 2014, the IEEE Trans. on Industrial Informatics Best Paper Award in 2012, the European Union FP7 Marie Curie RG Award in 2009.