scientific reports

Check for updates

OPEN

An energy-aware and Q-learning-based area coverage for oil pipeline monitoring systems using sensors and Internet of Things

Amir Masoud Rahmani[1], Saqib Ali[2], Mazhar Hussain Malik[3], Efat Yousefpoor[4], Mohammad Sadegh Yousefpoor[4], Amir Mousavi[5,6,10,11], Faheem khan[7]✉ & Mehdi Hosseinzadeh[8,9]✉

Pipelines are the safest tools for transporting oil and gas. However, the environmental effects and sabotage of hostile people cause corrosion and decay of pipelines, which bring financial and environmental damages. Today, new technologies such as the Internet of Things (IoT) and wireless sensor networks (WSNs) can provide solutions to monitor and timely detect corrosion of oil pipelines. Coverage is a fundamental challenge in pipeline monitoring systems to timely detect and resolve oil leakage and pipeline corrosion. To ensure appropriate coverage on pipeline monitoring systems, one solution is to design a scheduling mechanism for nodes to reduce energy consumption. In this paper, we propose a reinforcement learning-based area coverage technique called CoWSN to intelligently monitor oil and gas pipelines. In CoWSN, the sensing range of each sensor node is converted to a digital matrix to estimate the overlap of this node with other neighboring nodes. Then, a Q-learning-based scheduling mechanism is designed to determine the activity time of sensor nodes based on their overlapping, energy, and distance to the base station. Finally, CoWSN can predict the death time of sensor nodes and replace them at the right time. This work does not allow to be disrupted the data transmission process between sensor nodes and BS. CoWSN is simulated using NS2. Then, our scheme is compared with three area coverage schemes, including the scheme of Rahmani et al., CCM-RL, and CCA according to several parameters, including the average number of active sensor nodes, coverage rate, energy consumption, and network lifetime. The simulation results show that CoWSN has a better performance than other methods.

Today, wireless sensor networks (WSNs) have become an attractive research subject for many researchers in industry and university[1,2]. WSNs consist of a high number of sensor nodes distributed in the network environment to monitor the Region of Interest (RoI)[3,4]. Initially, these networks were specifically designed for military applications[5,6]. However, the growing advances in these networks have led to new technologies such as the Internet of Things (IoT)[7,8]. The IoT has created new opportunities to communicate things around us, such

[1]Future Technology Research Center, National Yunlin University of Science and Technology, Yunlin, Taiwan. [2]Department of Information Systems, College of Economics and Political Science, Sultan Qaboos University, Al Khoudh, Muscat, Oman. [3]HoD Computing and IT (CIT) Global College of Engineering and Technology, P.O Box 2546, CPO Ruwi 112, Muscat, Oman. [4]Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran. [5]Faculty of Civil Engineering, Technische Universitat Dresden, 01069 Dresden, Germany. [6]John von Neumann Faculty of Informatics, Obuda University, Budapest 1034, Hungary. [7]Artificial Intelligence Laboratory, Gachon University, Seongnam, Republic of Korea. [8]Mental Health Research Center, Psychosocial Health Research Institute, Iran University of Medical Sciences, Tehran, Iran. [9]Computer Science, University of Human Development, Sulaymaniyah, Iraq. [10]Institute of Information Engineering, Automation and Mathematics, Slovak University of Technology in Bratislava, Bratislava, Slovakia. [11]Institute of Information Society, University of Public Service, Budapest 1083, Hungary. ✉email: faheem@gachon.ac.kr; hosseinzadeh.m@iums.ac.ir

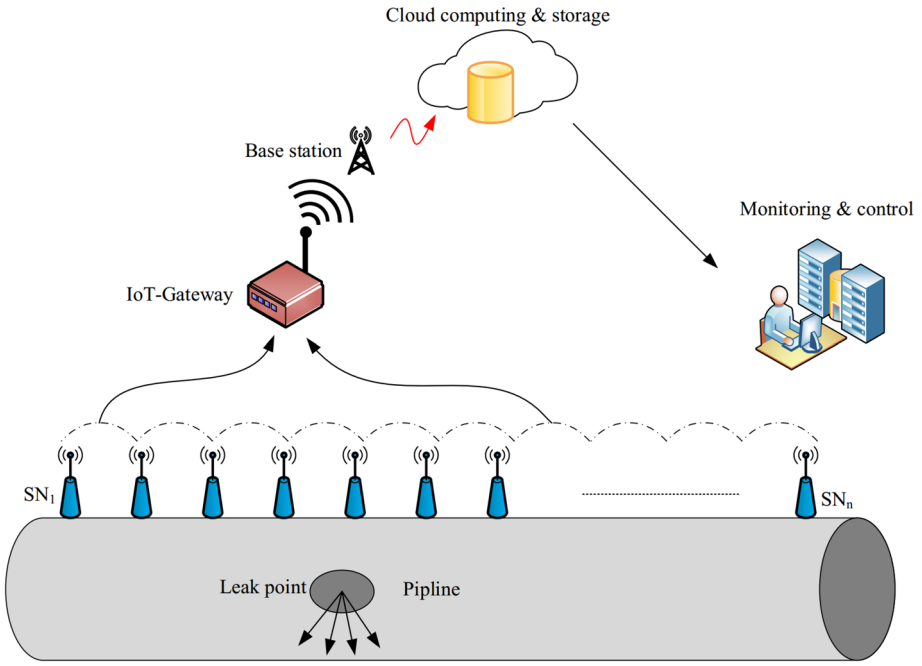

Figure 1. Smart oil pipeline.

as lamp switches[9], oil and gas transmission pipelines, industrial machines[10], home equipment[11], cars, and the human body using the Internet platform[12,13]. Today, this new technology is applied in various industrial and engineering fields such as pipeline monitoring systems. Intelligent monitoring on oil pipelines is a combination of industry and IoT[14,15]. In recent years, smart pipeline monitoring systems are possible due to advances in low-consumption electrical circuits and small, low-consumption, and cheap electronic equipment production, such as smart sensors[16,17]. Figure 1 shows a smart oil pipeline. The purpose of smart pipelines is to collect different information about the health of the pipeline using heterogeneous sensors. Because pipeline systems are responsible for transporting oil and gas, any leakage in the pipeline can cause financial and environmental damage. Today, only some of the important points are controlled in a pipeline. These points are several kilometers away from each other. This monitoring method is ineffective and inflexible[17,18]. A smart pipeline can provide a better understanding of the pipeline network. In large-scale pipelines, different sensor nodes are installed on the pipeline so that they can collect and process various parameters such as temperature, pressure, humidity, audio, and contamination and send this information to the control center.

The sensor nodes sense the target or phenomena occurred in their sensing ranges and process the data collected from this area and send the information to BS directly or using a multi-hop manner[19–21]. When monitoring pipelines, the main challenge is energy consumption because sensor nodes are deployed in the soil. This creates restrictions such as poor data transmission and data loss[22,23]. Also, this underground environment imposes major limitations on sensor nodes, especially poor radio frequency (RF) so that the RF transmission range in soil is significantly lower than in air[24,25]. Therefore, communication between nodes is much more limited in the underground environment. Therefore, we must consider shorter communication ranges for sensor nodes. Furthermore, repair or replacement of nodes is very costly. As a result, sensor nodes should have a long lifetime and consume less energy because they have a small communication and sensing range and limited energy resources. Therefore, providing proper coverage and maintaining connections play a very important role in efficiency and optimal performance of smart oil pipelines.

Coverage means the region or point monitored by sensor nodes scattered in that environment. Sensor nodes cover a region or point when that point or region is inside their sensing ranges[26,27]. Therefore, if a large number of sensor nodes are installed on the pipeline, this pipeline is covered properly and reliably. However, the pipeline structure is very complicated and dynamic. As a result, it is very difficult to measure all parameters related to the pipeline structure such as pressure and temperature to detect the pipeline corrosion. Therefore, the full coverage of the pipeline structure is impossible. Thus, in each period, only a number of nodes installed on the pipeline are activated to partially monitor the pipeline and provide a proper coverage rate. Coverage schemes can be implemented in two forms, including centralized and decentralized (distributed). In centralized coverage methods, only BS manages the coverage operation in the network. While in distributed coverage schemes, sensor nodes collaboratively execute coverage operations. Also, coverage methods can be executed statically and dynamically. Static coverage schemes determine the locations of sensor nodes in the network deterministically before bootstrapping the network. Note that this strategy does not change over the network lifetime. Dynamic methods periodically update the status of sensor nodes in the network. This means that this coverage strategy changes throughout the network lifetime.

On the other hand, network lifetime is a very important issue when covering the network because this parameter specifies how long time the pipeline network can work properly to meet the coverage requirements. Therefore, the network lifetime is an important criterion for evaluating smart pipelines. A suitable solution for this issue is to design a scheduling mechanism for sensor nodes. In this mechanism, in each scheduling period, only part of sensor nodes are activated in the network, and other nodes are in sleep status to reduce their energy consumption. The active sensor nodes should guarantee the desired partial coverage rate. The scheduling issue in the network can be considered as an optimization issue. However, real-world optimization issues are very complicated because they are large and dynamic. As a result, it is necessary to use computational intelligence-based methods such as reinforcement learning (RL) to solve these issues[28,29]. Today, RL algorithms are being popular rapidly because they can successfully find optimal response at a proper time. RL is suitable for solving issues such as routing, data aggregation, and coverage in WSN and IoT. RL is an appropriate tool in computational intelligence. It can learn optimal policy through interaction with the environment[30,31]. In this paper, we use the RL algorithm to achieve proper coverage rate so that our scheme uses the minimum number of active sensor nodes and improves energy consumption in the network.

In this paper, we propose an appropriate area coverage method called CoWSN to intelligently monitor oil and gas pipelines, so that CoWSN balances energy consumption and increases coverage quality in the network. The main contributions of CoWSN are as follows:

- In CoWSN, sensing range of each sensor node is converted to the digital matrix using a new, efficient, and distributed technique. The digital matrix helps us to calculate the overlap of this node with other neighboring nodes using geometric mathematics.
- In CoWSN, a Q-Learning-based scheduling mechanism is presented to calculate the activity time of sensor nodes based on three parameters, including the overlap between a sensor node and neighboring nodes, energy, and distance to BS.
- In CoWSN, the replacement time of nodes is predicted using an appropriate technique so that the data transmission process between sensor nodes and the base station is not disrupted.

In the following, the paper is organized as follows: in "Related works", the related works are expressed. Then, the basic concepts used in the proposed method are summarized in "Basic concepts". "System model" describes the system model in CoWSN. "Proposed schemeProposed scheme" explains our proposed method in detail. "Simulation and result evaluation" compares the simulation results of CoWSN with other coverage schemes. Finally, the conclusion of the paper is presented in "Conclusion".

Related works

In[32], the CCM-RL technique is presented in WSNs to maintain connections and coverage using reinforcement learning. CCM-RL tries to maximize coverage rate and maintain connections along with energy efficiency. In this method, nodes execute a learning algorithm to learn their optimal activity. As a result, CCM-RL activates a subset of nodes at a specific time. This reduces energy consumption in the network and provides an appropriate coverage rate and suitable connectivity. CCM-RL is a dynamic, distributed, and scalable coverage method. However, the scheduling mechanism has only taken attention to two parameters, including distance and coverage rate, and has ignored energy parameter. Also, CCM-RL has a lot of delay.

In[5], an area coverage approach based on fuzzy logic (FL) and shuffled frog-leaping algorithm (SFLA) is proposed. This approach balances energy consumption, increases network lifetime and improves coverage quality. This method calculates the overlap between sensor nodes using a distributed digital matrix-based approach. Then, a fuzzy scheduling mechanism is designed. This mechanism considers three parameters, including overlap, residual energy, and distance between each node and the base station to determine the activity time of each sensor node. Moreover, this method uses a strategy to predict the death of sensor nodes and prevent holes in network. Finally, this approach uses SFLA to find the best replacement strategy, which covers the holes created in the network and maximizes coverage rate. This coverage method is distributed, dynamic, and scalable. However, this method has a high communication overhead due to fuzzy scheduling mechanism. Also, the use of SFLA increases delay in the network.

In[33], the CCA technique is offered in two forms, including distributed and centralized, for homogeneous WSNs. CCA solves the *k*-coverage issue when deploying sensor nodes in the network. For solving this issue, CCA tries to use the lowest number of sensor nodes and increase network lifetime. CCA designs a scheduling process. According to this process, a subset of nodes is selected for covering the desired area. Centralized CCA can be implemented dynamically and statically. The dynamic method has a greater time complexity in comparison with the static method. Also, the dynamic approach has a better coverage rate than the static scheme. In general, distributed CCA is more scalable than centralized CCA because the distributed scheme relies on local information. Of course, distributed CCA has a lot of communication overhead.

In[34], a partial coverage technique based on learning automata (PCLA) is suggested in WSNs. PCLA solves the coverage issue and uses the minimum number of sensor nodes and maintains appropriate connectivity. This method uses learning automata (LA) to determine the activity time of nodes in the network. PCLA creates a backbone in the network. Therefore, a subset of the sensor nodes is selected for covering the RoI and maintaining connectivity. PCLA is a distributed, dynamic, and scalable method. However, this method has a lot of overhead.

In[35], a coverage method based on the genetic algorithm called MIGA is presented in heterogeneous WSNs. This scheme is inspired by IGA. MIGA presents a function to estimate the area coverage. However, this function is incomplete and does not consider the various overlap scenarios when overlapping two nodes. MIGA includes five components: individual representation, population initialization, genetic operators, fitness function, and

VFA optimization. MIGA achieves a high-quality response for maximizing coverage rate. However, MIGA is a centralized coverage method. It has a lot of computational overhead and is not scalable.

In[36], the maximum coverage sets scheduling (MCSS) mechanism is presented in WSN. MCSS schedules the coverage sets and improves network lifetime. This method uses a greedy algorithm for searching the problem. MCSS has acceptable time complexity and computational complexity. MCSS assumes that coverage sets and time slots of nodes are predetermined. MCSS is a centralized method and is not scalable. It takes into account only the activity time of nodes and does not consider other parameters such as energy and distance.

In[37], a barrier coverage method is offered in homogeneous wireless sensor networks. It uses the minimum number of sensor nodes when covering the network. In this method, the authors calculate the overlap between sensor nodes based on the angle between their sensing ranges. Furthermore, this method detects failed nodes and covers the holes created in the network. The most important advantage of this method is the proper coverage rate with the lowest sensor nodes. However, the authors have only taken attention to two parameters, including distance and overlap. Also, this method cannot predict the death time of nodes in the network. This scheme increases the latency in the network.

In[38], two coverage schemes are proposed for heterogeneous wireless sensor networks. These schemes use the improved cuckoo search (ICS) algorithm and chaotic flower pollination algorithm (CFPA). These methods try to reduce the implementation cost and energy consumption in the network. This method presents a fitness function, which considers only one parameter, including the overlap between nodes. It can be improved by considering more parameters. ICS and CFPA have a small computational complexity. They are simple and fast (high convergence speed) and can achieve a high-quality response. Furthermore, these schemes are static. This means that they explore the best replacement for sensor nodes and this strategy is fixed over network life.

In[39], two area coverage schemes are suggested in WSN. These approaches utilize the genetic algorithm (GA) and particle swarm optimization (PSO). The authors assume that the network consists of a number of obstacles. Then, they define the coverage issue in this network and use GA and PSO for addressing it. The important advantage of these approaches is to obtain the highest coverage rate with suitable computational overhead. However, this method considers only one parameter (i.e. the overlap between sensor nodes) when designing the cost function and does not consider other parameters such as energy. Also, this method focuses only on maximum area coverage and does not consider network lifetime. It is centralized and static. This reduces scalability.

In[40], a mathematical model is proposed to solve the coverage issue in WSN. This method moves sensor nodes toward low-density network areas to maximize the coverage rate in the network. This method distributes the sensor nodes in the network evenly. It uses an improved version of the virtual force algorithm. This method has low computational complexity. However, it is static and centralized and is not scalable. In this method, the goal is to maximize area coverage and does not pay attention to network lifetime.

Basic concepts

In this section, we briefly describe a well-known reinforcement learning technique called Q-Learning because we use this technique in the proposed method for designing the scheduling mechanism.

Q-Learning. Reinforcement learning (RL) allows machines or agents to learn their ideal behavior in a particular situation based on previous experience[28]. A RL-based model learns through interaction with the environment and collects information to do a specific activity. Q-Learning is a model-free and off-policy reinforcement algorithm. Q-Learning helps one agent to learn its optimal actions. According to this learning algorithm, state-action pairs are stored in a table called Q-table. This table receives a state-action pair as input and returns the Q-value as output. In Q-Learning, the goal is to maximize the Q-value. To achieve this goal, the agent adjusts its action strategy according to the reward received from the environment's feedback. In the learning process, the agent evaluates how many an action is suitable in the current state to choose a better action in the next iteration. Q-value is updated in each iteration using Eq. (1):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \tag{1}$$

where t is current iteration, $a \in A$ is the action set, r_{t+1} indicates the reward value received by the agent after doing the action a_t in the state s_t. When the agent performs the action a_t, its state changes from s_t to s_{t+1}. $\max_a Q(s_{t+1}, a)$ indicates maximum Q value when the agent performs the action a in the next iteration. $0 < \alpha \leq 1$ is the learning rate. If $\alpha = 0$, then the agent does not learn anything. If $\alpha = 1$, the agent learns only the last experience. In the proposed method, we consider $\alpha = 0.1$. Also, $0 < \gamma \leq 1$ indicates the discount factor. It represents the reward importance. In fact, it indicates the agent's effort for discovering the environment. We consider $\gamma = 0.7$ in the proposed method. Note that there are two techniques, including ϵ-greedy and Boltzmann in reinforcement learning to create a balance between exploration and exploitation[28]. In ϵ-greedy, the quantitative allocation strategy with a small value ϵ is used for exploring. In the proposed method, we used the ϵ-greedy technique to create a balance between exploration and exploitation.

System model

This section consists of four subsections: network model, energy model, sensing model, and communication model. In the following, we describe each subsection in detail.

Network model. In the proposed method, we consider a heterogeneous network with N sensor nodes. The nodes are heterogeneous. This means that they have different energy resources, sensing ranges, and communi-

cation ranges. They are randomly distributed in the network environment. Sensor nodes are equipped with a positioning system. Therefore, they are aware of their spatial coordinates (x_i, y_i) in the network. Also, the location of the base station (x_{BS}, y_{BS}) is known for each node in the network. Each sensor node knows its remaining energy ($E_{residual}$) at any moment. If sensor nodes are in communication ranges of each other, they can directly communicate with each other through a wireless communication channel. In this model, the network includes one base station, N_{Static} static sensor nodes, and $N_{Dynamic}$ mobile sensor nodes. Where,

$$N_{Static} + N_{Dynamic} = N \tag{2}$$

And,

$$N_{Dynamic} \ll N_{Static} \tag{3}$$

In the following, we describe the task of each node:

- Base station (BS): This node is responsible for receiving and processing information of sensor nodes.
- Static sensor nodes: These nodes are responsible for sensing the RoI and sending the sensed data to the base station.
- Mobile sensor nodes: These nodes are responsible for covering holes caused by the death of sensor nodes in the network.

Energy model. In CoWSN, when a transmitter node such as SN_i sends its data (k bits) to a receiver node like SN_j and the distance between the two nodes is equal to d. SN_i calculates the energy consumed for sending k bits according to Eq. (4):

$$E_{TX}(k,d) = \begin{cases} E_{elec} \times k + E_{fs} \times k + d^2, & d < d_0 \\ E_{elec} \times k + E_{mp} \times k + d^4, & d \geq d_0 \end{cases} \tag{4}$$

Also, SN_j calculates the energy consumed for receiving k bits using Eq. (5):

$$E_{RX}(k,d) = E_{elec} \times k \tag{5}$$

where E_{elec} indicates the energy used by transmitter/receiver circuit, E_{fs} and E_{mp} are the energy required for the transmitter amplifier in the free space and multipath models, respectively. Equation (6) computes d_0, which is the threshold of transmission distance:

$$d_0 = \sqrt{\frac{E_{fs}}{E_{mp}}} \tag{6}$$

Sensing model. CoWSN uses the binary sensing model that is also called 0/1 model. In this model, each SN_i with spatial coordinates (x_i, y_i) can sense the circular area. The radius of this area is equal to RS_i. Consider one point in the RoI, for example $P = (x_p, y_p)$. In binary sensing model, SN_i can sense P only when their Euclidean distance is lower than radius RS_i. In this case, we state that this point is inside the sensing range of SN_i. Otherwise, P is outside the sensing range of SN_i and cannot be covered by this node. This issue is expressed in Eq. (7):

$$C(S,P) = \begin{cases} 1 & d(SN_i,P) \leq RS_i \\ 0 & d(SN_i,P) > RS_i \end{cases} \tag{7}$$

And, the distance between SN_i and P is shown by $d(SN_i, P)$. This parameter is calculated by Eq. (8):

$$d(SN_i,P) = \sqrt{(x_i - x_p)^2 + (y_i - y_p)^2} \tag{8}$$

Communication model. CoWSN uses the binary disk model as the communication model. This model is similar to the sensing model. According to the communication model, the communication radius (RC) is known as the upper communication bound. This means that if there are two nodes that are in the communication ranges of each other, they can communicate directly together. Note that communication radius (RC) is greater than sensing radius (RS), so that:

$$RS < RC \tag{9}$$

Proposed scheme

Our scheme (CoWSN) is an area coverage technique. It increases the coverage quality, balances energy consumption in the network, and improves network lifetime. CoWSN consists of three parts:

- Converting sensing ranges of sensor nodes to digital matrix
- Q-learning-based scheduling mechanism
- Node replacement

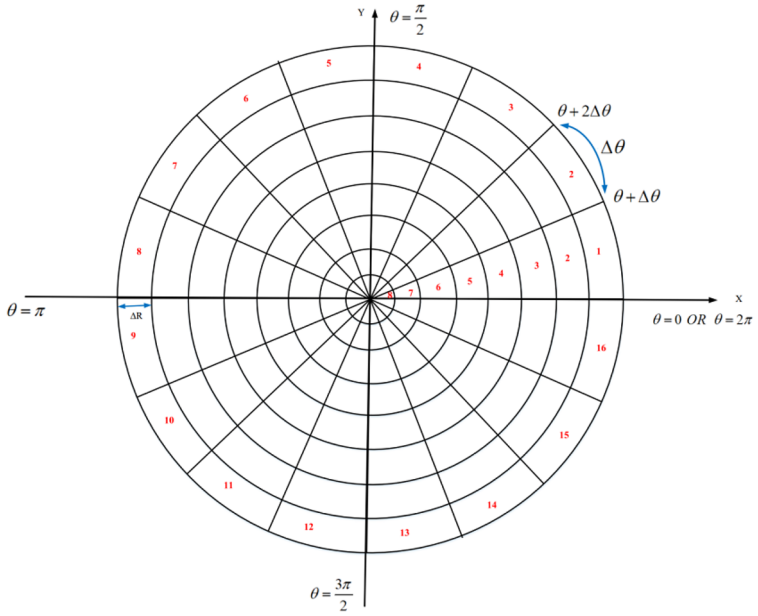

Figure 2. Dividing the sensing range of a sensor node into small rectangular sections.

Converting sensing ranges of sensor nodes to digital matrix. In this section, a decentralized technique is introduced for calculating the overlap of sensor nodes with their neighbors. According to this technique, the sensing range of a node is converted into a digital matrix. Note that a matrix is known as a digital matrix if all elements are zero or one. In the following, we describe this process in detail.

First, the sensing range of a node (for example, SN_i) is converted into a digital matrix. In this process, the node's coordinates is considered as the pole in the polar coordinate system and the sensing range of this node is displayed as a circular area C_i with the sensing radius RS_i. C_i is divided into n sectors ($\widehat{sec}_p,\ p = 1, 2, \ldots, n$) and m smaller circles ($c_q,\ q = 1, 2, \ldots, m$) with the same center. So that, $n = \frac{2\pi}{\Delta\theta}$ and $m = \frac{RS_i}{\Delta R}$. Also, $\Delta\theta$ is the angle of each $\hat{sec}_p$. Furthermore, each circle c_q has a radius such as r_q, this process is shown in Eqs. (10) and (11):

$$C_i = \begin{cases} \widehat{sec}_1 : \theta \le \hat{sec}_1 \le \theta + \Delta\theta \\ \widehat{sec}_2 : \theta + \Delta\theta \le \hat{sec}_2 \le \theta + 2\Delta\theta \\ \vdots \\ \widehat{sec}_n : \theta + (n-1)\Delta\theta \le \hat{sec}_n \le \theta + n\Delta\theta \end{cases} \tag{10}$$

$$C_i = \begin{cases} c_1 : r_1 = m\Delta R \\ c_2 : r_2 = (m-1)\Delta R \\ \vdots \\ c_m : r_m = \Delta R \end{cases} \tag{11}$$

Figure 2 shows an example in which C_i is divided into 16 sectors and 8 smaller circles. According to Fig. 2, this process partitions C_i into small rectangular sections. Note that $\Delta\theta$ and ΔR can be adjusted based on the problem requirements. When the user selects $\Delta\theta$ and ΔR close to zero, the result will be more accurate. However, this work increases memory consumption.

Now, C_i is converted to an $m \times n$ digital matrix. In this matrix, rows and columns are equal to smaller circles (c_q) and sectors ($\widehat{sec}_p,\ p = 1, 2, \ldots, n$), respectively. Furthermore, matrix elements (a_{qp}) represent rectangular sections, so that $q = 1, \ldots, m$ and $p = 1, \ldots, n$. Note that a_{qp} can be one or zero. In fact, a_{qp} is equal to one when the corresponding rectangular section overlaps with the sensing range at least one neighboring node. Otherwise, a_{qp} is equal to zero. Note that a_{qp} will be zero, when the corresponding rectangular section is not fully covered by neighboring nodes. An example of this digital matrix is shown in Fig. 3.

In the following, we describe how to determine the value of a_{qp}. In the first step, each sensor node like SN_i shares its information, including its identifier (ID_i), spatial coordinates (x_i, y_i), remaining energy ($E_{residual_i}$) and sensing radius (RS_i), with its own neighbors and broadcasts a Hello message for them. Then, the node stores the information of its neighbors in a table called $Table_{neighbor}$, which is shown in Table 1.

Now, SN_i can obtain the Euclidean distance between itself and its neighbors such as SN_j according to Eq. (12):

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{12}$$

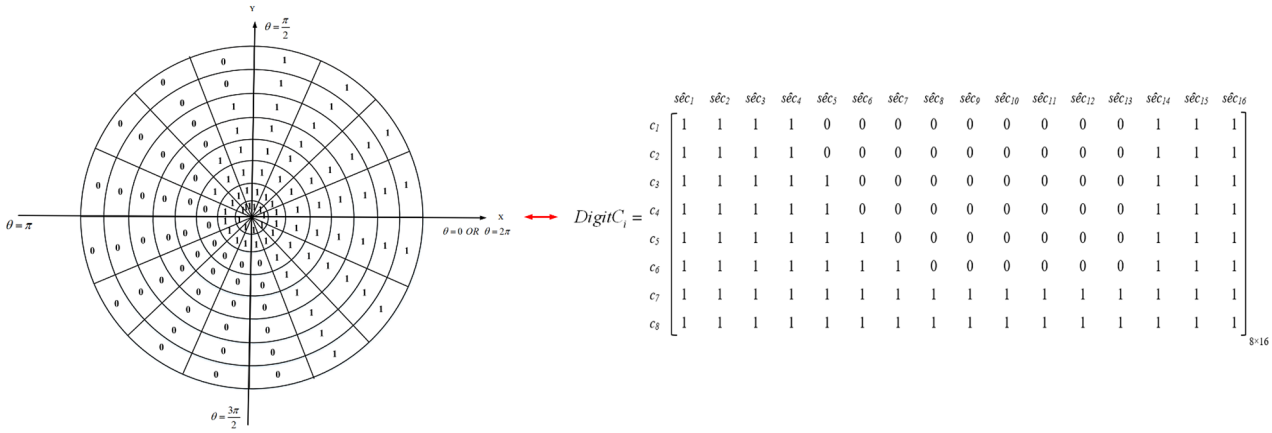

Figure 3. Digital matrix corresponding to the sensing range of the sensor node.

Number	ID	Spatial coordinates	Sensing radius	Remaining energy	Scheduling state
1	ID_j	(x_j, y_j)	RS_j	$E_{residual_j}$	$T_{Scheduling_j}$
2	ID_k	(x_k, y_k)	RS_k	$E_{residual_k}$	$T_{Scheduling_k}$

Table 1. $Table_{neighbor}$ stored in SN_i.

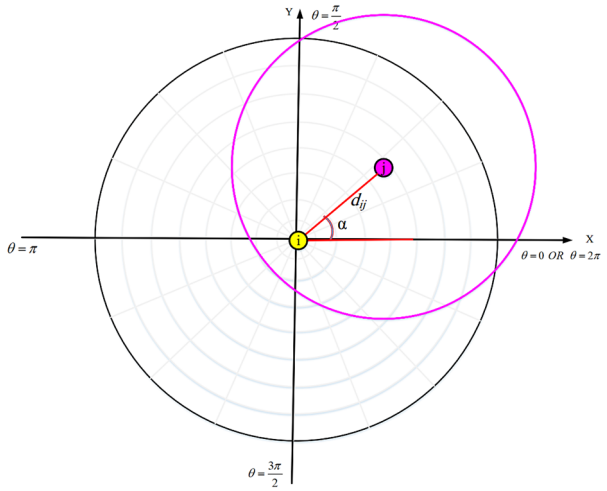

Figure 4. The angle of SN_j with regard to SN_i.

So that (x_i, y_i) and (x_j, y_j) are coordinates SN_i and SN_j, respectively.

If $d_{ij} \leq RS_j - RS_i$ or $RS_i - RS_j < d_{ij} < RS_i + RS_j$, then two nodes overlap in their sensing range. This means that if $d_{ij} \leq RS_j - RS_i$ then all a_{qp} where, $q = 1, \ldots, m$ and $p = 1, \ldots, n$, will be one. Otherwise, if $RS_i - RS_j < d_{ij} < RS_i + RS_j$, then the angle of C_j with regard to the pole (i.e. SN_i) in the polar coordinate system is calculated using Eq. (13):

$$\alpha = \arctan\left(\frac{x_j - x_i}{y_j - y_i}\right), \quad 0 \leq \alpha \leq 2\pi \tag{13}$$

See Fig. 4.

For obtaining the value of a_{qp} in the digital matrix and calculating the overlap between SN_i and SN_j, we follow the following commands.

- If $d_{ij} \geq R_j + r_q$, so that $1 \leq q \leq 8$ in this example, c_q and all smaller circles are outside C_j. As a result, a_{qp} will be zero in the corresponding rows.

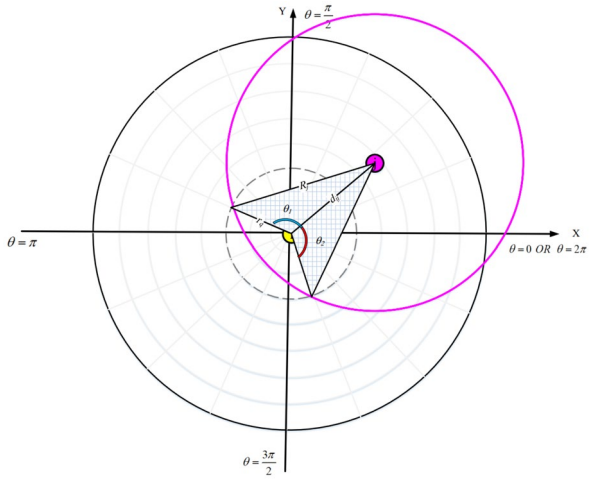

Figure 5. Overlapping area between c_q and C_j.

- If $d_{ij} \leq R_j - r_q$, so that $1 \leq q \leq 8$ in this example, c_q and all smaller circles are inside C_j. Therefore, a_{qp} will be one in the corresponding rows.
- If $R_j - r_q < d_{ij} < R_j + r_q$, c_q and C_j partially overlap with each other, and this overlap is calculated as follows:
  - As shown in Fig. 5, a triangle with three vertices, (x_i, y_i), (x_j, y_j), and the intersection point of c_q and C_j is considered. Now, compute the length of three sides of this triangle.
  - Here, the angle $\theta_1 = \theta_2$ is obtained using the cosine law:

$$R_j^2 = r_q^2 + d_{ij}^2 - 2r_q d_{ij} \cos\theta_1 \tag{14}$$

where,

$$\theta_1 = \arccos\left(\frac{r_q^2 + d_{ij}^2 - R_j^2}{2r_q d_{ij}}\right), \quad 0 \leq \theta_1 \leq \pi \tag{15}$$

  - Equation (16) computes the overlapping area (γ_q) between c_q and C_j:

$$\alpha - \theta_1 \leq \gamma_q \leq \alpha + \theta_1, \qquad 0 \leq \gamma_k \leq 2\pi \tag{16}$$

  - Now, Eq. (17) computes a_{qp} corresponding to the row c_q and the sector $\hat{sec}_p$.

$$a_{qp} = \begin{cases} 1\ , & IF\ R_j - r_q < d_{ij} < R_j + r_q\ AND\ \alpha - \theta_1 \leq \hat{sec}_p \leq \alpha + \theta_1 \\ 0\ , & otherwise \end{cases} \tag{17}$$

This process is repeated for all $c_q, 1 \leq q \leq 8$ to calculate all a_{qp}, where, $q = 1, \ldots, m$ and $p = 1, \ldots, n$. Algorithm 1 presents the pseudocode of this process.

Algorithm 1 Converting sensing range to digital matrix

Input: $\Delta\theta$, ΔR, m, n,
N {N is the number of sensor nodes in the network.}
M {M is number of neighbors of SN_i}
SN_i: $DigitC_i$, (x_i, y_i), RS_i, C_i
SN_j: (x_j, y_j), RS_j, C_j
Output: $DigitC_i$
Begin
for $i = 1$ to N **do**
  SN$_i$: Obtain its spatial coordinates using GPS;
  SN$_i$: Broadcast a *HELLO message* for its neighbors;
  SN$_i$: Receive *HELLO messages* from its neighbors;
  SN$_i$: Update $Table_{neighbor}$ according to these messages;
end for
for $\eta = 1$ to M **do**
  SN$_i$: Calculate Euclidean distance between SN_i and SN_η;
  if $(d_{i,\eta} < RS_i + RS_\eta)$ **then**
    SN$_i$: Compute the angle between SN_i and SN_η according to Eq. 13;
    for $q = 1$ to m **do**
      if $(d_{i,\eta} \geq RS_\eta + r_q)$ **then**
        for $p = 1$ to n **do**
          $a_{qp} = 0$;
        end for
      else if $(d_{i,\eta} \leq RS_\eta - r_q)$ **then**
        for $p = 1$ to n **do**
          $a_{qp} = 1$;
        end for
      else if $(RS_\eta - r_q < d_{i,\eta} < RS_\eta + r_q)$ **then**
        SN$_i$: Calculate the angle θ_1 based Eq. 15;
        for $p = 1$ to n **do**
          if $(\alpha - \theta_1 \leq s\hat{e}c_p \leq \alpha + \theta_1)$ **then**
            $a_{qp} = 1$;
          else
            $a_{qp} = 0$;
          end if
        end for
      end if
    end for
  end if
end for
End

Q-learning-based scheduling mechanism. In this section, our goal is to design a Q-learning-based scheduling mechanism so that each sensor node learns independently and automatically the best ON/OFF time slots in any scheduling round ($T_{Scheduling}$) to maximize coverage rate and network lifetime. The learning process immediately begins after deploying sensor nodes in the network. At the start time ($t = 0$), we initialize the Q-learning parameters, including learning rates (α), discount factor (γ), and Q value. Also, all sensor nodes are activated at $t = 0$. Then, the time slots are updated and modified in the learning process according to the Q-learning algorithm to achieve optimal response. As stated in "Converting sensing ranges of sensor nodes to digital matrix", at $t = 0$, each sensor node shares its information such as its identifier (ID_i), spatial coordinates (x_i, y_i), remaining energy ($E_{residual_i}$), the scheduling state ($T_{Scheduling}$), and sensing radius RS_i with its own neighbors, and stores their information in $Table_{neighbor}$ shown in Table 2. In the learning process, this information is used.

In the following, we describe various components in this scheduling mechanism:
Agent In this protocol, each sensor node (SN_i) plays the agent role.
Environment In this learning issue, the network plays the role of environment.

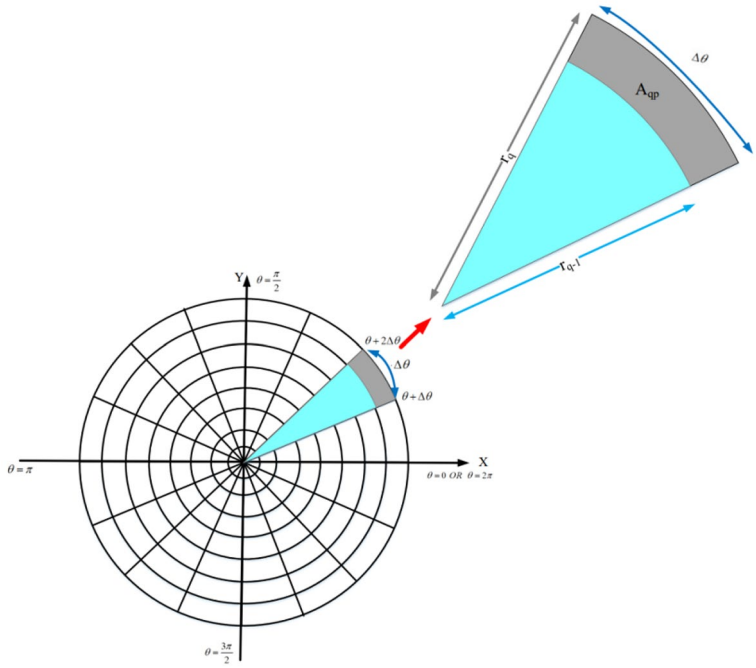

Figure 6. The calculation of A_{qp} area.

State In this issue, the state of an agent is the overlap value of this agent with other active neighbors. The overlapping area (O_i) of each node is calculated using the digital matrix. First, the rectangular area A_{qp} (Gray area shown in Fig. 6) is calculated based on Eq. (18):

$$A_{qp} = Area_{Circle\ sector_{qp}} - Area_{Circle\ sector_{(q-1)p}} \tag{18}$$

where $Area_{Circle\ sector_{qp}}$ is the sector area p in the circle c_q. It is calculated through Eq. (19):

$$Area_{Circle\ sector_{qp}} = \frac{1}{2} r_q^2 \Delta\theta \tag{19}$$

After merging Eqs. (19) and (18), we have:

$$A_{qp} = \frac{1}{2} r_q^2 \Delta\theta - \frac{1}{2} r_{q-1}^2 \Delta\theta = \frac{1}{2} \Delta\theta \left(r_q^2 - r_{q-1}^2 \right) \tag{20}$$

As stated in Eq. (11):

$$\begin{aligned} r_q &= \big(m - (q-1)\big)\Delta R \\ r_{q-1} &= \big(m - (q-2)\big)\Delta R \end{aligned} \tag{21}$$

As a result, A_{qp} is equal to Eq. (22):

$$\begin{aligned} A_{qp} &= \frac{1}{2} \Delta\theta\, \Delta R^2 \left(\big(m - (q-1)\big)^2 - \big(m - (q-2)\big)^2 \right) \\ &= \frac{1}{2} \Delta\theta\, \Delta R^2 \big(2(q-m) - 3\big) \end{aligned} \tag{22}$$

Therefore, O_i is obtained according to Eq. (23):

$$O_i = \frac{\sum_{q=1}^{m} \sum_{p=1}^{n} \frac{1}{2} (a_{qp}) \Delta\theta\, \Delta R^2 \big(2(q-m) - 3\big)}{\pi R S_i^2} \tag{23}$$

where $m \times n$ is the size of $DigitC_i$ and RS_i represents the sensing radius of SN_i.

Action In the scheduling issue, the action is a set of all activities that can be done by the agent SN_i in the state O_i. Assume that the scheduling round ($T_{Scheduling}$) includes ten time slots ($Ts_i, \quad i = 1, \ldots, 10$). Each sensor node can be active in some of these time slots (i.e. $Ts_i = T_{ON}$) or it can be sleep in other time slots ($Ts_i = T_{OFF}$) so that $T_{Scheduling} = \sum_{i=1}^{10} Ts_i$. To better understand this issue, consider the example presented in Table 2. The purpose of the learning algorithm is to find the best possible scheduling for each sensor node SN_i. As a result, the action corresponding to SN_i is as $T_{Scheduling}^t = [Ts_1, Ts_2, \ldots, Ts_{10}]^t$ at the iteration t.

$T_{Scheduling}$									
Ts_1	Ts_1	Ts_1	Ts_1	Ts_1	Ts_1	Ts_1	Ts_1	Ts_1	Ts_1
T_{OFF}	T_{OFF}	T_{ON}	T_{ON}	T_{ON}	T_{OFF}	T_{OFF}	T_{ON}	T_{ON}	T_{ON}

Table 2. Time slots corresponding to SN_i.

Award The award indicates the environment's feedback with regard to the action performed by the agent SN_i in the state O_i. If this action ($T_{Scheduling}$) is successful, the environment has positive feedback. Otherwise, it has negative feedback. In CoWSN, we consider two parameters for calculating the award function: the remaining energy ($E_{residual}$) and the distance between each node and the BS (D_{i-BS}). The reason for choosing these parameters is that high-energy nodes receive a positive award from the environment and increase their Q-value to stay in the ON mode for more time slots. Also, low-energy nodes receive a negative reward and decrease their Q-value to do their activities in fewer time slots. As a result, we choose the energy parameter to balance energy consumption in the network. Also, the purpose of choosing the distance parameter is that the sensor nodes close to the BS receive a positive reward from the environment and increase their Q-value to stay in the ON mode at more time slots because these nodes do more operations than other nodes. Therefore, the award function is calculated using Eq. (24):

$$r\left(O_i^t, T_{Scheduling}^t\right) = \begin{cases} r_{\min}, & \frac{4}{5} \le O_i^{t+1} \le 1 \\ \left(\omega\left(1 - \frac{\sqrt{(x_i - x_{BS})^2 + (y_i - y_{BS})^2} - \mu_d}{\sigma_d}\right) + (1-\omega)\left(\frac{E_{residual} - \mu_e}{\sigma_e}\right)\right), & \frac{2}{5} < O_i^{t+1} < \frac{4}{5} \\ r_{\max}, & 0 \le O_i^{t+1} \le \frac{2}{5} \end{cases} \tag{24}$$

In Eq. (24), (x_i, y_i) and (x_{BS}, y_{BS}) are the spatial coordinates of SN_i and BS, respectively. μ_d indicates the average distance of active neighboring nodes to the BS obtained from $Table_{neighbor}$ of SN_i. Furthermore, σ_d is the distance standard deviation of active neighbors at the current iteration. Also, $E_{residual}$ indicates the remaining energy of SN_i. μ_e and σ_e are the average energy of active neighbors and the energy standard deviation of active neighbors at the current iteration, respectively. ω is a weight coefficient.

$$\mu_d = \frac{1}{m}\sum_{j=1}^{m}\sqrt{(x_j - x_{BS})^2 + (y_j - y_{BS})^2} \tag{25}$$

$$\sigma_d = \sqrt{\frac{1}{m}\sum_{j=1}^{m}\left(\sqrt{(x_j - x_{BS})^2 + (y_j - y_{BS})^2} - \mu_d\right)^2} \tag{26}$$

$$\mu_e = \frac{1}{m}\sum_{j=1}^{m} E_{residual_j} \tag{27}$$

$$\sigma_e = \sqrt{\frac{1}{m}\sum_{j=1}^{m}\left(E_{residual_j} - \mu_e\right)^2} \tag{28}$$

In Eqs. (25), (26), (27) and (28), m is the number of active neighboring nodes of SN_i at the current iteration.

Convergence condition It is the time interval required by the learning algorithm to achieve the optimal response. In CoWSN, if the learning algorithm finds that the response does not change in the five last iterations, then the algorithm is convergent. Finally, the optimal response (the scheduling determined for SN_i) is stored.

In algorithm 2, the pseudo-code of the scheduling mechanism is presented.

Algorithm 2 Scheduling mechanism based on Q-learning

Input: (x_i, y_i), (x_{BS}, y_{BS}), $E_{residual_i}$, $DigitC_i$
Output: $T_{Scheduling}$
Begin
SN$_i$: Initialize all Q-learning parameters such as Q-table, learning rate, discount factor, and reward function;
SN$_i$: Start the learning process based on Q-learning algorithm;
while the convergence condition is not met **do**
  SN$_i$: Calculate O_i using Eq. 23;
  if $\frac{4}{5} \leq O_i^{t+1} \leq 1$ **then**
    $r\left(O_i^t, T_{Scheduling}^t\right) = r_{\min}$
  else if $\frac{2}{5} < O_i^{t+1} < \frac{4}{5}$ **then**
    $r\left(O_i^t, T_{Scheduling}^t\right) = \omega\left(1 - \frac{\sqrt{(x_i - x_{BS})^2 + (y_i - y_{BS})^2} - \mu_d}{\sigma_d}\right) + (1-\omega)\left(\frac{E_{residual} - \mu_e}{\sigma_e}\right)$
  else if $0 \leq O_i^{t+1} \leq \frac{2}{5}$ **then**
    $r\left(O_i^t, T_{Scheduling}^t\right) = r_{\max}$
  end if
  SN$_i$: Perform an appropriate action ($T_{Scheduling}^{t+1}$) based on the current state (O_i^{t+1});
  SN$_i$: Update Q-table using the Q-learning system;
end while
Return $T_{Scheduling}^t$;
End

Node replacement. After launching the network, the sensor nodes begin their activities according to the time slots specified in the scheduling round. These activities lead to energy loss of sensor nodes. In this section, we predict the death time of nodes to prevent possible holes in the network. This work does not allow that the data transmission process between nodes and the BS is disrupted. To achieve this purpose, each sensor node (for example, SN_i) updates periodically its $Priority_i$ based on Eq. (29) to determine its importance for replacing.

$$Priority_i = \left(1 - \frac{O_i}{\pi RS_i^2}\right) + \left(\frac{Packet_{size_i}}{Buffer_{size_i}}\right) \tag{29}$$

where $\left(1 - \frac{O_i}{\pi RS_i^2}\right)$ is the non-overlapping area of SN_i. Note that O_i is the overlapping area of SN_i obtained through Eq. (13). Also, RS_i is the sensing radius of SN_i. $\left(\frac{Packet_{size_i}}{Buffer_{size_i}}\right)$ calculates the data traffic in SN_i. $Packet_{size_i}$ is the number of packets in the buffer of SN_i at a specific time. Also, $Buffer_{size_i}$ represents the buffer size of SN_i.

When SN_i loses its energy so that its energy is lower than a threshold. This node sends a warning message along with its $Priority_i$ to the BS. Then, the BS compares $Priority_i$ with $P_{Threshold}$ (a threshold value for priority) to decide on the replacement of this node. Note that $P_{Threshold}$ is a constant amount so that $P_{Threshold} > 0$.

- If $Priority_i > P_{Threshold}$ is larger than $P_{Threshold}$, then SN_i has a higher importance for replacing because if SN_i dies in the network, then the normal network operations are damaged. Therefore, the BS sends a Coverage message to the mobile node closest to SN_i for replacing this node.
- If $Priority_i$ is smaller or equal to $P_{Threshold}$, then the death of SN_i cannot disrupt the normal network operations. Therefore, the BS ignores SN_i.

Algorithm 3 presents the pseudocode of the node replacement.

Parameter	Value
Simulator	NS-2.35
Network size	$1000 \times 1000\ m^2$
Total number of nodes	250–2000
Simulation time	1200 s
Sensing radius	25, 30, and 35 m
Communication radius	50, 60, and 70 m
Initial energy of nodes	100 J
Energy consumed by active nodes	57 mA
Energy consumed by inactive nodes	0.40 μA

Table 3. Simulation parameters.

Algorithm 3 Replacing sensor nodes

Input: O_i, $Packet_{size_i}$, $Buffer_{size_i}$
Output: Coverage a hole
Begin
SN_i: Compute $Priority_i$ using Eq. 29;
if $E_{residual_i} < Treshold$ **then**
  SN_i: Forward a *warning message* to the BS;
end if
BS: Receive the *warning message*;
BS: Extract $Priority_i$ from the *warning message*;
if $Priority_i > P_{Threshold}$ **then**
  BS: Send a *Coverage message* to the nearest mobile node;
end if
End

Simulation and result evaluation

In this section, we simulate CoWSN with NS2 to evaluate its performance. The simulation results are compared with the scheme of Rahmani et al.[5], CCM-RL[32], and CCA[33]. We assume that the network includes 250–2000 heterogeneous sensor nodes, which are randomly distributed in the network. The nodes have different sensing ranges (i.e. 25, 30, and 35 m) and communication ranges (i.e. 50, 60, and 70 m). The network size is equal to $1000 \times 1000\ m^2$. When nodes are active, they consume energy equal to 57mA. Also, when nodes are inactive, they consume 0.40 μA. Table 3 presents simulation parameters in summary. We evaluate the performance of CoWSN in terms of four parameters, including the average number of active sensor nodes, coverage rate, energy consumption, and network lifetime.

The average number of active sensor nodes. The number of active sensor nodes indicates the subsets of the active nodes selected for covering the Region of Interest (RoI). As shown in Fig. 7, CoWSN has the best performance in terms of the number of active nodes at a scheduling period. This means that CoWSN lowers the number of active nodes by 7.67%, 11.04%, and 13.32% compared to Rahmani et al., CCM-RL, and CCA, respectively. This is because CoWSN uses a Q-Learning-based scheduling mechanism to determine the activity time of the sensor nodes. CoWSN and Rahmani et al. calculate the overlap between a sensor node and its neighbors by a precise approach. Although, the scheme of Rahmani et al. has a fuzzy scheduling mechanism for calculating the activity time of nodes in the network. This method has a weaker performance than our method. Also, CCM-RL focuses only on the distance parameter when calculating the overlap of nodes. This is not a precise method and can have a lot of error. In addition, CCA does not present any approach to calculate the overlap between nodes. On the other hand, CoWSN considers two parameters, including energy and distance to the base station, to determine the best scheduling for sensor nodes. CCA focuses on the energy parameter in the scheduling process, but does not pay attention to the overlap between nodes. Furthermore, CCM-RL does not consider energy of nodes in the learning process. It is an important weakness of this method. According to Fig. 7, when the number of nodes is increasing in the network, all methods increase the average number of active nodes. Note that a coverage method cannot activate all nodes at all times because they consume high energy and die quickly, which reduces network lifetime. As a result, when the density of nodes is low in the network, the active nodes cannot cover the whole RoI. Thus, the coverage rate is reduced. But when the density of nodes is increasing, each method increases the number of active nodes. Therefore, the coverage quality of the RoI is improved.

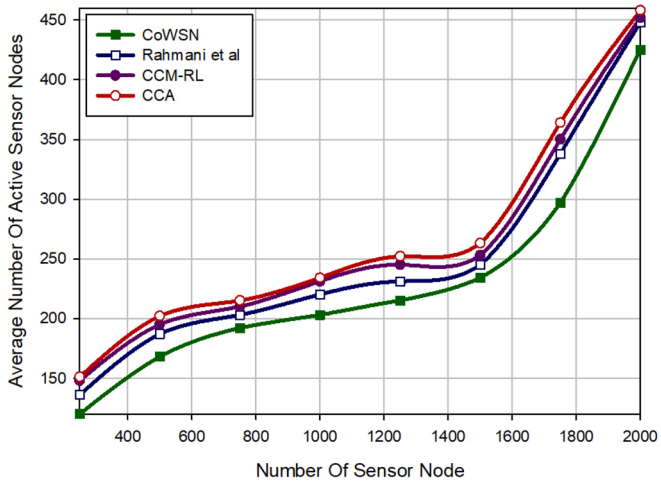

Figure 7. Comparison of the average number of active nodes in different schemes.

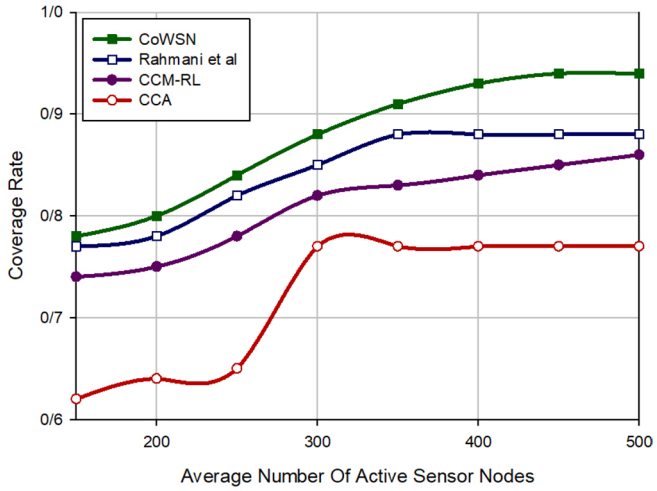

Figure 8. Comparison of coverage rate in different schemes.

Coverage rate. The coverage rate is defined as the percentage of the RoI covered by active nodes. As shown in Fig. 8, CoWSN increases the coverage rate by 4.15%, 8.50%, and 21.87% in comparison with Rahmani et al., CCM-RL, and CCA, respectively. This is due to the fact that our method calculates overlapping between nodes accurately and penalizes the nodes with more overlapping. This means that they receive the lowest reward to be in the sleep mode for more slot times. This helps CoWSN to achieve the best coverage rate with the lowest active nodes in the network. Moreover, CoWSN can predict the death of sensor nodes and timely replace them to prevent coverage quality loss. Meanwhile, CCM-RL and CCA do not provide any approach to replace dead nodes. Although, the scheme of Rahmani et al. addresses this issue using SFLA. According to Fig. 8, when the number of active nodes is more than 350, CoWSN achieves a coverage rate more than 90%, which is very desirable. While the scheme of Rahmani et al. has achieved a coverage rate equal to 88% for this number of active nodes. This coverage rate is fixed and does not change when increasing the number of active nodes. In CCM-RL, the coverage rate is not constant and is improved when increasing the number of active nodes. Although, in the best mode, CCM-RL has reached a coverage rate equal to 86%. In CCA, the coverage rate is equivalent to 77% for 350 active nodes or more.

Energy consumption. As shown in Fig. 9, CoWSN reduces energy consumption by 27.27%, 51.51%, and 70.19% compared to Rahmani et al., CCM-RL, and CCA, respectively. This is because our Q-learning-based scheduling mechanism considers the energy parameter when designing the reward function. In our method, high-energy nodes receive a reward and increase their Q value to be in the active mode for more time. Moreover, low-energy nodes are penalized to do their activities in a short time. Also, the scheme of Rahmani et al. considers the energy parameter in the scheduling process of sensor nodes. However, this method has high communica-

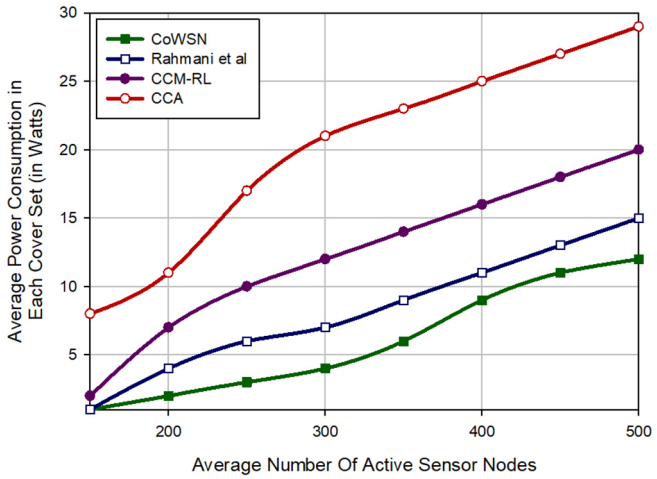

Figure 9. Comparison of the average energy consumption in different schemes.

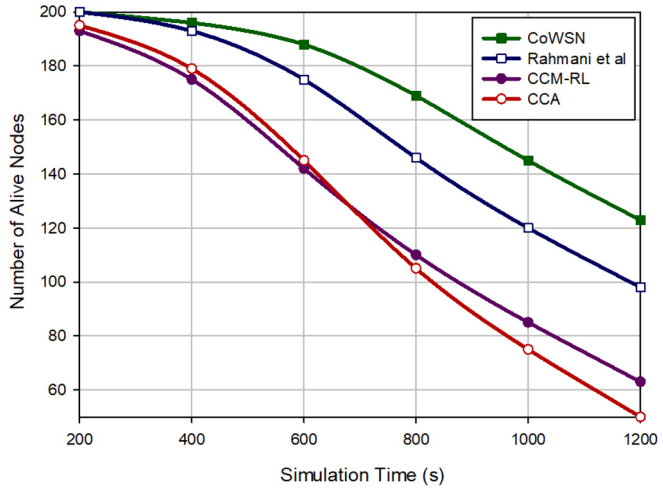

Figure 10. Comparison of network lifetime in different methods.

tion overhead because it uses fuzzy logic in the scheduling mechanism. Furthermore, it uses SFLA to cover the hole created in the network. SFLA increases energy consumption in this method. CCM-RL has the third rank in terms of energy consumption compared to other methods. This scheme does not consider the energy parameter in the scheduling process. Although, it uses the sensing range customization mechanism, which helps CCM-RL to consume energy efficiently. CCA has the worst performance in terms of energy consumption because it has high communication overhead.

Network lifetime. Figure 10 compares various methods in terms of network lifetime. We assume that there are 200 alive nodes in the network when doing this experiment. The nodes consume their energy over time. CoWSN improves the network lifetime by 9.55%, 32.94% and 36.32% in comparison with Rahmani et al., CCM-RL, and CCA, respectively. We described the reasons for this issue in "Energy consumption". CoWSN tries to evenly distribute energy consumption between sensor nodes in the network because it takes into account the energy parameter in the scheduling process.

Conclusion

In this paper, we presented an area coverage scheme called CoWSN to intelligently monitor gas and oil pipelines. The purpose of CoWSN is to reduce energy consumption, improve network lifetime, and achieve the highest coverage rate in the network. To achieve these goals, we used a digital matrix-based technique to calculate the overlap between each sensor node and its neighboring nodes. Then, we designed a Q-Learning-based scheduling mechanism to determine the activity time of each sensor node. Also, CoWSN uses a suitable strategy to timely detect the death of nodes and prevent holes in the network. To evaluate CoWSN, it is simulated with NS2 and

compared with the scheme of Rahmani et al., CCM-RL, and CCA methods. Simulation results show the successful performance of CoWSN. In this paper, we have used the WSN platform (and not a real-time gas or oil pipeline network) to simulate our scheme. In the future research direction, we try to evaluate our method in real gas or oil pipeline environments and under more scenarios so that the performance of our method is further identified. Also, we seek to improve the efficiency of our method using other machine learning (ML) techniques and evolutionary algorithms (EAs) in the future.

Received: 13 January 2022; Accepted: 25 April 2022
Published online: 10 June 2022

References

1. Yousefpoor, E., Barati, H. & Barati, A. A hierarchical secure data aggregation method using the dragonfly algorithm in wireless sensor networks. *Peer Peer Netw. Appl.*https://doi.org/10.1007/s12083-021-01116-3 *(2021).*
2. Yousefpoor, M. S. & Barati, H. Dynamic key management algorithms in wireless sensor networks: A survey. *Comput. Commun.* **134**, 52–69. https://doi.org/10.1016/j.comcom.2018.11.005 (2019).
3. Yousefpoor, M. S. *et al.* Secure data aggregation methods and countermeasures against various attacks in wireless sensor networks: A comprehensive review. *J. Netw. Comput. Appl.* **20**, 103118. https://doi.org/10.1016/j.jnca.2021.103118 (2021).
4. Yousefpoor, M. S. & Barati, H. DSKMS: A dynamic smart key management system based on fuzzy logic in wireless sensor networks. *Wirel. Netw.* **26**(4), 2515–2535. https://doi.org/10.1007/s11276-019-01980-1 (2020).
5. Rahmani, A. M. *et al.* An area coverage scheme based on fuzzy logic and shuffled frog-leaping algorithm (SFLA) in heterogeneous wireless sensor networks. *Mathematics* **9**(18), 2251. https://doi.org/10.3390/math9182251 (2021).
6. Lee, S. W. *et al.* An energy-aware and predictive fuzzy logic-based routing scheme in flying ad hoc networks (FANETS). *IEEE Access* **9**, 129977–130005. https://doi.org/10.1109/access.2021.3111444 (2021).
7. Zakariayi, S. & Babaie, S. DEHCIC: A distributed energy-aware hexagon based clustering algorithm to improve coverage in wireless sensor networks. *Peer Peer Netw. Appl.* **12**(4), 689–704. https://doi.org/10.1007/s12083-018-0666-9 (2019).
8. Shahraki, A., Taherkordi, A., Haugen, Ø. & Eliassen, F. Clustering objectives in wireless sensor networks: A survey and research direction analysis. *Comput. Netw.* **180**, 107376. https://doi.org/10.1016/j.comnet.2020.107376 (2020).
9. Shahraki, A., Taherkordi, A. & Haugen, Ø. TONTA: Trend-based online network traffic analysis in ad-hoc IoT networks. *Comput. Netw.* **194**, 108125. https://doi.org/10.1016/j.comnet.2021.108125 (2021).
10. Kuscu, M., Ramezani, H., Dinc, E., Akhavan, S. & Akan, O. B. Fabrication and microfluidic analysis of graphene-based molecular communication receiver for Internet of Nano Things (IoNT). *Sci. Rep.* **11**(1), 1–20. https://doi.org/10.1038/s41598-021-98609-1 (2021).
11. Alammari, A., Moiz, S. A. & Negi, A. Enhanced layered fog architecture for IoT sensing and actuation as a service. *Sci. Rep.* **11**(1), 1–23. https://doi.org/10.1038/s41598-021-00926-y (2021).
12. Dampage, U., Bandaranayake, L., Wanasinghe, R., Kottahachchi, K. & Jayasanka, B. Forest fire detection system using wireless sensor networks and machine learning. *Sci. Rep.* **12**(1), 1–11. https://doi.org/10.1038/s41598-021-03882-9 (2022).
13. Alsamhi, S. H. *et al.* Green internet of things using UAVs in B5G networks: A review of applications and strategies. *Ad Hoc Netw.* **20**, 102505. https://doi.org/10.1016/j.adhoc.2021.102505 (2021).
14. Gil-Castiñeira, F., Costa-Montenegro, E. & Silva, J. S. Reliable link level routing algorithm in pipeline monitoring using implicit acknowledgements. *Sensors* **21**(3), 968. https://doi.org/10.3390/s21030968 (2021).
15. Wadhaj, I., Thomson, C., & Ghaleb, B. An RPL based optimal sensors placement in pipeline monitoring WSNs. In *International Conference on Emerging Technologies and Intelligent Systems* (pp. 533–546) (Springer, 2021). https://doi.org/10.1007/978-3-030-85990-9_43.
16. Varshney, S., Kumar, C. & Swaroop, A. Pipeline surveillance along the international border using hybrid optimization algorithm. *J. Ambient Intell. Human. Comput.*https://doi.org/10.1007/s12652-021-02934-2 *(2021).*
17. Singh, R. *et al.* Zigbee and long-range architecture based monitoring system for oil pipeline monitoring with the internet of things. *Sustainability* **13**(18), 10226. https://doi.org/10.3390/su131810226 (2021).
18. Khalifeh, A. *et al.* Wireless sensor networks for smart cities: Network design, implementation and performance evaluation. *Electronics* **10**(2), 218. https://doi.org/10.3390/electronics10020218 (2021).
19. Roy, S., Mazumdar, N. & Pamula, R. An energy and coverage sensitive approach to hierarchical data collection for mobile sink based wireless sensor networks. *J. Ambient. Intell. Humaniz. Comput.* **12**(1), 1267–1291. https://doi.org/10.1007/s12652-020-02176-8 (2021).
20. Gao, K. *et al.* Linear system design with application in wireless sensor networks. *J. Ind. Inf. Integrat.* **20**, 100279. https://doi.org/10.1016/j.jii.2021.100279 (2021).
21. Das, S. K. & Kapelko, R. On the range assignment in wireless sensor networks for minimizing the coverage-connectivity cost. *ACM Trans. Sens. Netw.* **17**(4), 1–48. https://doi.org/10.1145/3457408 (2021).
22. Wang, S., You, H., Yue, Y. & Cao, L. A novel topology optimization of coverage-oriented strategy for wireless sensor networks. *Int. J. Distrib. Sens. Netw.* **17**(4), 1550147721992298. https://doi.org/10.1177/1550147721992298 (2021).
23. Mysorewala, M. F., Cheded, L. & Aliyu, A. Review of energy harvesting techniques in wireless sensor-based pipeline monitoring networks. *Renew. Sustain. Energy Rev.* **157**, 112046. https://doi.org/10.1016/j.rser.2021.112046 (2022).
24. Sadeghioon, A. M., Metje, N., Chapman, D. N. & Anthony, C. J. SmartPipes: Smart wireless sensor networks for leak detection in water pipelines. *J. Sens. Actuator Netw.* **3**(1), 64–78 (2014).
25. Aalsalem, M. Y., Khan, W. Z., Gharibi, W., Khan, M. K. & Arshad, Q. Wireless sensor networks in oil and gas industry: Recent advances, taxonomy, requirements, and open challenges. *J. Netw. Comput. Appl.* **113**, 87–97 (2018).
26. Yi, L. *et al.* Reinforcement-learning-enabled partial confident information coverage for IoT-based bridge structural health monitoring. *IEEE Internet Things J.* **8**(5), 3108–3119. https://doi.org/10.1109/JIOT.2020.3028325 (2020).
27. El-Hosseini, M., ZainEldin, H., Arafat, H. & Badawy, M. A fire detection model based on power-aware scheduling for IoT-sensors in smart cities with partial coverage. *J. Ambient. Intell. Humaniz. Comput.* **12**(2), 2629–2648. https://doi.org/10.1007/s12652-020-02425-w (2021).
28. Jang, B., Kim, M., Harerimana, G. & Kim, J. W. Q-learning algorithms: A comprehensive classification and applications. *IEEE Access* **7**, 133653–133667. https://doi.org/10.1109/ACCESS.2019.2941229 (2019).
29. Rahmani, A. M. *et al.* Machine learning (ML) in medicine: Review, applications, and challenges. *Mathematics* **9**(22), 2970. https://doi.org/10.3390/math9222970 (2021).
30. Sarwar, S. *et al.* Reinforcement learning based adaptive duty cycling in LR-WPANs. *IEEE Access* **8**, 161157–161174. https://doi.org/10.1109/ACCESS.2020.3021016 (2020).
31. Ge, J. *et al.* Q-learning based flexible task scheduling in a global view for the Internet of Things. *Trans. Emerg. Telecommun. Technol.* **32**(8), e4111. https://doi.org/10.1002/ett.4111 (2021).

32. Sharma, A. & Chauhan, S. A distributed reinforcement learning based sensor node scheduling algorithm for coverage and connectivity maintenance in wireless sensor network. *Wirel. Netw.* **26**(6), 4411–4429. https://doi.org/10.1007/s11276-020-02350-y (2020).
33. Yu, J., Wan, S., Cheng, X. & Yu, D. Coverage contribution area based *k*-coverage for wireless sensor networks. *IEEE Trans. Veh. Technol.* **66**(9), 8510–8523. https://doi.org/10.1109/TVT.2017.2681692 (2017).
34. Mostafaei, H., Montieri, A., Persico, V. & Pescapé, A. A sleep scheduling approach based on learning automata for WSN partial-coverage. *J. Netw. Comput. Appl.* **80**, 67–78. https://doi.org/10.1016/j.jnca.2016.12.022 (2017).
35. Hanh, N. T., Binh, H. T. T., Hoai, N. X. & Palaniswami, M. S. An efficient genetic algorithm for maximizing area coverage in wireless sensor networks. *Inf. Sci.* **488**, 58–75. https://doi.org/10.1016/j.ins.2019.02.059 (2019).
36. Luo, C. *et al.* Maximizing network lifetime using coverage sets scheduling in wireless sensor networks. *Ad Hoc Netw.* **98**, 102037. https://doi.org/10.1016/j.adhoc.2019.102037 (2020).
37. Benahmed, T. & Benahmed, K. Optimal barrier coverage for critical area surveillance using wireless sensor networks. *Int. J. Commun Syst* **32**(10), e3955. https://doi.org/10.1002/dac.3955 (2019).
38. Binh, H. T. T., Hanh, N. T., Van Quan, L. & Dey, N. Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks. *Neural Comput. Appl.* **30**(7), 2305–2317. https://doi.org/10.1007/s00521-016-2823-5 (2018).
39. Binh, H. T. T., Hanh, N. T., Nghia, N. D. & Dey, N. Metaheuristics for maximization of obstacles constrained area coverage in heterogeneous wireless sensor networks. *Appl. Soft Comput.* **86**, 105939. https://doi.org/10.1016/j.asoc.2019.105939 (2020).
40. Li, Q. & Liu, N. Monitoring area coverage optimization algorithm based on nodes perceptual mathematical model in wireless sensor networks. *Comput. Commun.* **155**, 227–234. https://doi.org/10.1016/j.comcom.2019.12.040 (2020).

Author contributions

A.M.R., M.S.Y., and E.Y.: initial conceptualization. S.A.: experimental setup. A.M., M.H.: field testing A.M., M.H., M.H.M., F.K., A.M.R., M.S.Y., and E.Y. wrote the main manuscript text. M.S.Y., and E.Y.: prepared figures. M.S.Y., M.H., S.A., F.K.: wrote the analysis section. M.H. and A.M.: reviewed the final manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to F.k. or M.H.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.